

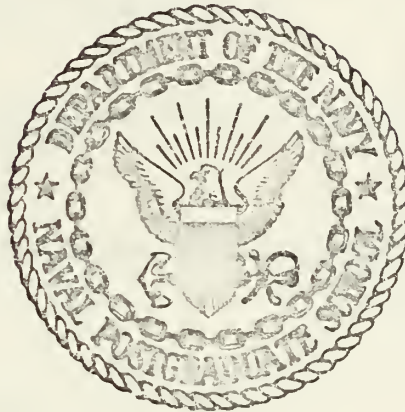
EFFECTIVE METHODS FOR SOLUTION OF  
NONLINEAR REACTOR DYNAMICS  
PROBLEMS USING FINITE  
ELEMENTS

Richard Allen Olsen

DUDLEY KNOX LIBRARY  
NAVAL POSTGRADUATE SCHOOL  
MONTEREY, CALIFORNIA 93940

# NAVAL POSTGRADUATE SCHOOL

## Monterey, California



# THESIS

Effective Methods for Solution of Nonlinear Reactor  
Dynamics Problems Using Finite Elements

by

Richard Allen Olsen

December 1975

Thesis Advisor:  
Co-Advisor:

D.H. Nguyen  
D. Salinas

Approved for public release; distribution unlimited.

T170813



UNCLASSIFIED

 KNOX LIBRARY  
 NAVAL POSTGRADUATE SCHOOL  
 MONTEREY, CALIFORNIA 93940

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

## REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS  
BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)

 Effective Methods for Solution of Nonlinear  
 Reactor Dynamics Problems Using Finite  
 Elements

 5. TYPE OF REPORT & PERIOD COVERED  
 Masters' Thesis; December  
 1975

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

Richard Allen Olsen

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS

 Naval Postgraduate School  
 Monterey, CA 93940
10. PROGRAM ELEMENT, PROJECT, TASK  
AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

 Naval Postgraduate School  
 Monterey, CA 93940

12. REPORT DATE

December 1975

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME &amp; ADDRESS (if different from Controlling Office)

 Naval Postgraduate School  
 Monterey, CA 93940

15. SECURITY CLASS. (of this report)

Unclassified

15a. DECLASSIFICATION/DOWNGRADING  
SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The solution of the nonlinear two-dimensional reactor dynamics equation subjected to prompt feedback conditions using the finite element technique leads to the matrix formulation  $A_{ij}\psi_j = B_{ij}\psi_j + C_{ijk}\psi_j\psi_k$  ( $i, j, k = 1, \dots, N$ ).

This system has been solved directly in a previous work; but because the nonlinearity  $C_{ijk}\psi_j\psi_k$  was premultiplied by  $[A]^{-1}$ , large computer storage

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)



was required for the small problem considered. The task of this thesis is the development of computational techniques which allow the problem to be solved for large systems. Specifically, these techniques are: (1) the treatment of the nonlinearity on the element level, (2) the compacting of the sparse matrices to include only non-zero terms, and (3) the construction of a new computer code based on the Crank-Nicolson formulation for the solution of differential equations.

To support the theory presented, test problems were solved by the original method, the linearized technique, and the Crank-Nicolson treatment. The results were analyzed and compared graphically. All three of the innovations developed in this thesis appear to be useful tools for solving nonlinear time dependent differential equations.







Effective Methods for Solution of Nonlinear Reactor  
Dynamics Problems Using Finite Elements

by

Richard Allen Olsen  
Lieutenant Commander, United States Navy  
B.S., United States Naval Academy, 1966

Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL  
December 1975



# ABSTRACT

The solution of the nonlinear two-dimensional reactor dynamics equation subjected to prompt feedback conditions using the finite element technique leads to the matrix formulation  $A_{ij}\dot{\psi}_j = B_{ij}\psi_j + C_{ijk}\psi_j\psi_k$  ( $i, j, k = 1, \dots, N$ ). This system has been solved directly in a previous work; but because the nonlinearity  $C_{ijk}\psi_j\psi_k$  was premultiplied by  $[A]^{-1}$ , large computer storage was required for the small problem considered. The task of this thesis is the development of computational techniques which allow the problem to be solved for large systems. Specifically, these techniques are: (1) the treatment of the nonlinearity on the element level, (2) the compacting of the sparse matrices to include only non-zero terms, and (3) the construction of a new computer code based on the Crank-Nicolson formulation for the solution of differential equations.

To support the theory presented, test problems were solved by the original method, the linearized technique, and the Crank-Nicolson treatment. The results were analyzed and compared graphically. All three of the innovations developed in this thesis appear to be useful tools for solving nonlinear time dependent differential equations.



## TABLE OF CONTENTS

I.	INTRODUCTION -----	8
II.	REDUCTION OF THE NONLINEARITY -----	14
	A. GLOBAL TREATMENT OF THE NONLINEARITY -----	14
	B. TREATMENT ON THE ELEMENT LEVEL -----	14
III.	SOLUTION PROCEDURE WITH THE LINEARIZED FORM -----	19
	A. GENERAL TREATMENT OF THE MATRIX EQUATION -----	19
	B. PRACTICAL CONSIDERATIONS WITH THE LINEARIZED TECHNIQUE ---	20
	C. DIFFICULTIES WITH THE APPROXIMATION -----	21
IV.	THE $N \times p$ COMPACTING SCHEME -----	22
	A. ESTABLISHED COMPACTING METHODS -----	22
	B. BASIS FOR THE COMPACTING SCHEME -----	23
	C. CONSTRUCTION OF THE $N \times p$ ARRAY -----	24
V.	THE CRANK-NICOLSON FORMULATION -----	26
VI.	SUMMARY OF AVAILABLE METHODS -----	30
VII.	TEST PROBLEMS AND RESULTS -----	31
	A. THE PHYSICAL MODEL -----	31
	B. COMPUTER PROCESSING CONSIDERATIONS -----	31
	C. PROBLEM ANALYSIS -----	31
	D. EVALUATION OF ERROR -----	34
VIII.	CONCLUSION -----	35
	APPENDIX A - TABLES -----	36
	APPENDIX B - FIGURES -----	39
	APPENDIX C - COMPUTER PROGRAMMING CODES -----	61
	LIST OF REFERENCES -----	120
	INITIAL DISTRIBUTION LIST -----	121



# LIST OF FIGURES

1.	REACTOR MODEL WITH 38 NODES -----	40
2.	NODAL CONNECTIVITY -----	41
3.	REACTOR MODEL WITH 132 NODES -----	42
4.	PLOT OF CENTER DISTURBANCE - CENTER POINT -----	43
5.	COMPARISON - CENTER DISTURBANCE - CENTER POINT -----	44
6.	PLOT OF CENTER DISTURBANCE - CORE POINT -----	45
7.	COMPARISON - CENTER DISTURBANCE - CORE POINT -----	46
8.	PLOT OF CENTER DISTURBANCE - REFLECTOR POINT -----	47
9.	COMPARISON - CENTER DISTURBANCE - REFLECTOR POINT -----	48
10.	PLOT OF UNIFORM DISTURBANCE - CENTER POINT -----	49
11.	COMPARISON - UNIFORM DISTURBANCE - CENTER POINT -----	50
12.	PLOT OF UNIFORM DISTURBANCE - CORE POINT -----	51
13.	COMPARISON - UNIFORM DISTURBANCE - CORE POINT -----	52
14.	PLOT OF UNIFORM DISTURBANCE - REFLECTOR POINT -----	53
15.	COMPARISON - UNIFORM DISTURBANCE - REFLECTOR POINT -----	54
16.	PLOT OF SKEW DISTURBANCE - CENTER POINT -----	55
17.	COMPARISON - SKEW DISTURBANCE - CENTER POINT -----	56
18.	PLOT OF SKEW DISTURBANCE - CORE POINT -----	57
19.	COMPARISON - SKEW DISTURBANCE - CORE POINT -----	58
20.	PLOT OF SKEW DISTURBANCE - REFLECTOR POINT -----	59
21.	COMPARISON - SKEW DISTURBANCE - REFLECTOR POINT -----	60





## ACKNOWLEDGEMENTS

With grateful appreciation I wish to acknowledge the persistent and dedicated efforts of my thesis advisors, Associate Professors D. H. Nguyen and D. Salinas.

I sincerely thank my many friends on the C.W. Church Computer Programming staff who contributed significant advice and direction toward the successful construction of my projects. Paramount in this group was Mr. Roger Hilleary whose mathematical expertise coupled with his detailed knowledge of available software clarified my thinking and provided the necessary insight for the task at hand.

Frequently there exists a noticable gap between the knowing how and the getting it done. The people who were able to help me close this gap were the Computer Center Operators, a really fine group of professionals. Among them was Mr. Ed Donnellan, a very patient man who led me through many intricacies of the job control language. To these individuals I owe much gratitude.

I appreciate my opportunity to study for the past two-and-a-half years at the Naval Postgraduate School. I thank those professors with whom I have worked for their contribution to my total educational experience. This experience includes not only an increased technical expertise but also a deeper maturity and a broader horizon. Among others, the names of T. Sarpkaya, J. Brock and T. Cooper will long stand out in my memory.



## I. INTRODUCTION

When temperature-dependent feedback is considered in the nuclear reactor dynamics problem, a nonlinear field equation in space and time results [Ref. 1]. For the non-homogeneous, or multi-region reactor, however, the space-dependent dynamics behavior following a nonlinear initial disturbance is no longer reachable in analytical form. Fortunately, by modeling the reactor as a system of finite regions of interest where the neutronic properties are known, the method of finite elements can be applied to yield the solutions. The fundamental concepts relating reactor behavior to the finite element formulation have been presented in 1974 [Ref. 2], but a more recent work by Nguyen and Salinas [Ref. 3] gives a thorough discussion of the complete problem. That work forms the basis and starting point for this thesis, the objective of which is to develop improved computational methods for dealing with that finite element formulation.

In Reference 3 the dynamic flux equation under prompt feedback conditions was given as

$$\frac{1}{V_m} \frac{\partial \psi_m(r, z, t)}{\partial t} = D_m \nabla^2 \psi_m + \gamma_m \Sigma_{am} \psi_m - \alpha_m K_m \Sigma_{am} \psi_m^2 \quad (1)$$

for each non-homogeneous zone "m" contained in the reactor body, where the usual symbols are employed:

$$\lambda_m = v \Sigma_{fm} / \Sigma_{am} - 1$$

$$K_m = e \Sigma_{fm} / \rho_m C_{pm} \quad (^\circ\text{C/unit flux-sec})$$

$$\gamma_m = (A_m / V_m) (h_m / \rho_m C_{pm}) \quad \text{sec}^{-1}$$



$\rho C_p$  = heat capacity

$\alpha$  = reactivity temperature coefficient  $^{\circ}\text{C}^{-1}$

$h$  = convection heat transfer coefficient

$A/V$  = heat transfer area to volume of energy  
generation ratio

$\Sigma_f$  = neutron fission cross-section

$\Sigma_a$  = neutron absorption cross-section

$\nu$  = neutrons emitted per fission

$e$  = energy produced per fission

The subscript "m" will be omitted from further discussion for simplicity.

The transformation of this general equation into a problem in finite dimensional vector space begins by constructing the following  $N$  term approximation<sup>1</sup>

$$\psi(\underline{x}, t) \approx \tilde{\psi}(\underline{x}, t) = \sum_{j=1}^N \psi_j(t) G_j(\underline{x}) \quad (2)$$

where  $N$  is the number of degrees of freedom, or nodes, in the space, and the  $G_j$  are the basis functions of the approximate solution space. The Galerkin method seeks to make the residual

$$R(\underline{x}, t) = L \tilde{\psi} - f,$$

where  $L\tilde{\psi} = f$  is the field equation, orthogonal to each of the basis functions so that

---

<sup>1</sup>This discussion leading to the establishment of the matrix equations is essentially an abridgement of the development given in reference 3.





$$\int_S G_i(\underline{x}) R(\underline{x}, t) d\underline{x} = 0 \quad i = 1, 2, \dots, N. \quad (3)$$

For the nuclear reactor dynamics problem of Eq. (1), the residual becomes

$$R(\underline{x}, t) = \frac{\partial \psi^2}{\partial t} - V D \nabla^2 \tilde{\psi} - V \lambda \Sigma_a \tilde{\psi} + w \Sigma_a \tilde{\psi}^2 \quad (4)$$

with  $V \lambda \Sigma_a = w$ . Putting this expression into Eq. (3) and integrating by parts (a distinct advantage of Galerkin), the following coefficients emerge

$$A_{IJ} = \iint_S G_I G_J r dr dz \quad (5a)$$

$$B_{IJ} = \iint_S \left[ \frac{\partial G_J}{\partial r} \cdot \frac{\partial G_I}{\partial r} + \frac{\partial G_J}{\partial z} \cdot \frac{\partial G_I}{\partial z} \right] r dr dz \quad (5b)$$

$$C_{IJK} = \iint_S G_I G_J G_K r dr dz \quad (5c)$$

$$I, J, K = 1, \dots, N$$

in cylindrical coordinates where  $dS = r dr dz$ . This allows Eq. (1) to be written as follows for the case of uniform neutronic properties within each reactor region  $m$ .

$$\sum_{J=1}^N A_{IJ} \dot{\psi}_J = - V_I D_I \sum_{J=1}^N B_{IJ} \psi_J + V_I \lambda_I \Sigma_{aI} \sum_{J=1}^N A_{IJ} \psi_J - \quad (6)$$

$$w_I \sum_{J=1}^N \sum_{K=1}^N C_{IJK} \psi_J \psi_K \quad I = 1, 2, \dots, N$$



This becomes, upon combining constants, in Einstein summation convention, where there is no sum on an underlined repeated index,

$$\begin{aligned} A_{IJ} \dot{\psi}_J &= AB_{IJ} \psi_J - w_{\underline{I}} C_{\underline{I}JK} \psi_J \psi_K \\ \dot{\psi}_i &= f_i(\psi_j, t) \end{aligned} \quad I, J, K, = 1, \dots, N \quad (7a)$$

With boundary conditions, the system is now well posed and ready for solution. The equation-solver must be chosen with care, however, because this system of ordinary differential equations is both stiff and non-linear. "Stiff" is used here and throughout this thesis to denote a system which gives a large response to a small stimulus; in this case, the nature of the reactor dynamics problem predicts a large change in flux over a small change in time.<sup>2</sup> Current experience suggests Gear's predictor-corrector method [Ref. 4] as written in the computer programming code DVOGER [Ref. 5]. It requires only 1) the locus (value) of the points at a given time, 2) a routine to evaluate the instantaneous derivatives at any time and 3) a routine to evaluate the Jacobian ( $J = \partial f_i(\psi, t) / \partial \psi_j$ ). Based on this information, the program assumes a time step, which may be quite small (the minimum size being a function of the particular computer), and tries to fit a trial (predicted) solution for that time interval. The corrected (integrated) solution is then obtained by iteration until convergence is attained. If the predicted and corrected solutions fail to match within a specified error criterion, the time step is decreased and the process repeated. On the other hand, if "excessive" accuracy is attained, the next attempted time step will be increased. Gear's method in DVOGER is well suited to stiff non-linear systems. Also contained in DVOGER is the Adam's

---

<sup>2</sup>This definition is somewhat broader in scope than that used by many other authors.



method which parallels that of Gear except that the Jacobian is not computed. Lacking the additional information about the rate of change, the Adam's method must proceed much more cautiously with stiff systems and hence marches forward with exceedingly small time steps. Present experience confirms the caveat advice given by DVOGER that the Adam's method is not intended for "stiff" cases.

The difficulty that arises from acquiring solutions to Eq. (7), then, should not, and, in light of the present experience, does not result from DVOGER or any other acceptable equation solver, but is, rather, a function of matrix size, vis a vis the number of nodal points in the finite element approximation. The finite element modeling of large reactors or the close scrutiny of small ones is severely restricted, therefore, unless the number of nodal points can be raised significantly. As an example, the test case considered in this thesis consisted of only 38 nodes but, when processed directly from Eq. (7), required over 300K bytes of storage in the IBM 360/67 computer. This requirement came largely from the  $[A]^{-1}[C_{IJK}]$  matrix alone needing a size of  $4(38 \times 38 \times 38) = 219K$  bytes in single precision. Manipulations with this cubic were indeed quite burdensome. Since machine processing time is, other things being equal, dependent on size, the combined time and space requirements would prohibit the consideration of even moderate sized problems except on the largest computers. Additionally, the inversion of  $[A]$  in Eq. (7) is indicated in order to provide an explicit value of  $\dot{\psi}$  for the DVOGER routine. It is this difficulty of size, and with it processing time, to which the remainder of this thesis is devoted. Experience with the DVOGER routine is presented, and the author's equation-solver based on



a Crank-Nicolson formulation is discussed as the analysis of the size problem is developed. Table I shows clearly the influence of matrix size on computer storage requirements.





## II. REDUCTION OF THE NONLINEARITY

### A. GLOBAL TREATMENT OF THE NONLINEARITY

The evaluation of Eq. (7) generates the following global matrix system, where the subscripts indicate the matrix size required for N number of nodal points

$$\left[ A_{N \times N} \right] \left\{ \dot{\psi}_{N \times 1} \right\} = \left[ B_{N \times N} \right] \left\{ \psi_{N \times 1} \right\} + \left[ C_{N \times N \times N} \right] \left\{ \psi^2_{(N \times N) \times 1} \right\} \quad (8)$$

where  $\psi$  is the time response of the flux at each nodal point and  $\psi^2$  is the product  $\psi_i \psi_j$   $i, j = 1, 2, \dots, N$ . Solved in this form, the major limitation of the procedure was the large  $N^3$  size of the cubic array C, which is introduced as the result of the nonlinearity of Eq. (1). With this size requirement rapidly increasing for a finer mesh consisting of more nodal points,<sup>3</sup> the investigation of Reference 3 was limited to a small but practical N of 38.

### B. TREATMENT ON THE ELEMENT LEVEL

The cubic form of C need not appear if it can be shown that the nonlinear portion of Eq. (8) can be broken apart into the product of two linear components such that one component can be evaluated on the element level and the element contributions then summed into global form,

---

<sup>3</sup>In general, for a nonlinearity of order m, a finite element problem of N nodes will generate a matrix size of  $N^{m+1}$ .



resulting in a "regular"  $N \times N$  matrix  $C^*$  ready for multiplication by the remaining  $\psi$  also summed to the global level. Proof that this is indeed the case, i.e.,

$$\left[ C_{N \times (N \times N)} \right] \left\{ \psi_{(N \times N) \times 1}^2 \right\} = \left[ C_{N \times N}^* \right] \left\{ \psi_{N \times 1} \right\} \quad (9)$$

is outlined as follows:

Reference 3 established the element nonlinear term as

$$\iint_{A_e} \zeta_i (\psi_j \zeta_j) (\psi_k \zeta_k) dA = C_{ijk} \psi_j(t) \psi_k(t) \quad (10)$$

where the integral is over the element area  $A_e$ ,

$$C_{ijk} = \iint_{A_e} \zeta_i \zeta_j \zeta_k (r_e \zeta_e) dr dz, \quad i, j, k, e = 1, 2, 3 \quad (11)$$

and  $\zeta$  being the local triangular coordinates, an innovation of Felippa [Ref. 6]. The coefficients  $C_{ijk}$  form a  $3 \times 3 \times 3$  element array possessing a regularity which suggests a kind of "cubic symmetry." The evaluation of this integral results in the following expression

$$\left. \begin{aligned} C_{111} &= \gamma^e [24r_1 + 6(r_2 + r_3)] \\ C_{112} &= \gamma^e [6r_1 + 4r_2 + 3r_3] \\ C_{113} &= \gamma^e [6r_1 + 2r_2 + 4r_3] \\ C_{122} &= \gamma^e [4r_1 + 6r_2 + 2r_3] \\ C_{123} &= \gamma^e [2(r_1 + r_2 + r_3)] \\ C_{133} &= \gamma^e [4r_1 + 2r_2 + 6r_3] \\ C_{222} &= \gamma^e [6r_1 + 24r_2 + 6r_3] \\ C_{333} &= \gamma^e [6r_1 + 6r_2 + 24r_3] \end{aligned} \right\} \quad (12)$$



where

$$\gamma^e = \pi A_e / 180$$

and

$$C_{121} = C_{112} = C_{211}$$

$$C_{321} = C_{231} = C_{213} \text{ etc as in Ref. 3.}$$

Assuming that  $\psi^2$  may be broken apart as  $\psi^2 = \psi \cdot \phi$ , the LHS (left hand side) of Eq. (10) is rewritten as

$$\iint_{A_e} \zeta_i (\zeta_j \psi_j) (\zeta_k \phi_k) dA \quad i, j, k = 1, 2, 3 \quad (13)$$

or

$$2\pi \iint_{A_e} \zeta_K [\zeta_1 \zeta_2 \zeta_3] \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \end{bmatrix} [\zeta_1 \zeta_2 \zeta_3] \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} r \, dr \, dz \quad (14)$$

Expanding and collecting terms yields

$$\begin{aligned} 2\pi \iint_{A_e} \zeta_K \left[ \psi_1 \left\{ \phi_1 (r_1 \zeta_1^3 + r_2 \zeta_1^2 \zeta_2 + r_3 \zeta_1^2 \zeta_3) \right. \right. \\ + \phi_2 (r_1 \zeta_1^2 \zeta_2 + r_2 \zeta_1 \zeta_2^2 + r_3 \zeta_1 \zeta_2 \zeta_3) \\ + \phi_3 (r_1 \zeta_1^2 \zeta_3 + r_2 \zeta_1 \zeta_2 \zeta_3 + r_3 \zeta_1 \zeta_3^2) \Big\} \\ + \psi_2 \left\{ \phi_1 (r_1 \zeta_1^2 \zeta_2 + r_2 \zeta_1 \zeta_2^2 + r_3 \zeta_1 \zeta_2 \zeta_3) + \phi_2 (r_1 \zeta_1 \zeta_2^2 + r_2 \zeta_2^3 + r_3 \zeta_2^2 \zeta_3) \right. \end{aligned} \quad (15)$$





$$\begin{aligned}
& + \phi_3(r_1 \zeta_1 \zeta_2 \zeta_3 + r_2 \zeta_2^2 \zeta_3 + r_3 \zeta_2 \zeta_3^2) \Big\} \\
& + \psi_3 \left\{ \phi_1(r_1 \zeta_1^2 \zeta_3 + r_2 \zeta_1 \zeta_2 \zeta_3 + r_3 \zeta_1 \zeta_3^2) + \phi_2(r_1 \zeta_1 \zeta_2 \zeta_3 + r_2 \zeta_2^2 \zeta_3 + r_3 \zeta_2 \zeta_3^2) \right. \\
& \left. + \phi_3(r_1 \zeta_1 \zeta_3^2 + r_2 \zeta_2 \zeta_3^2 + r_3 \zeta_3^3) \right\} \Big] \, dr \, dz \qquad k = 1, 2, 3
\end{aligned}$$

Since the quantities  $\psi_i$  and  $\phi_j$  ( $i, j = 1, 2, 3$ ) depend only on time, they are unaffected by the integration. Further, the expressions inside the parentheses are given a unique name  $d_{ijk}$  to fix their position in relation to a particular  $\psi_i \phi_j$  for some  $k = 1, 2, 3$ . Now the above expression may be written as

$$2\pi \, \psi_i \phi_j \iint_A d_{ijk} \, dr \, dz . \quad (16)$$

Using the integration formula from Felippa [Ref. 6]

$$\iint_{A_e} \zeta_1^\ell \zeta_2^m \zeta_3^n \, dA = \frac{\ell!m!n!}{(\ell+m+n+2)!} \times 2A_e \quad (17)$$

where  $\ell, m, n$  are the exponents of  $\zeta$  in a specific  $d_{ijk}$ . The integration of (16) gives

$$(\pi A_e / 180) \, \psi_i \phi_j \, f_{ijk} \quad (18)$$

and  $f_{ijk} \times \pi A_e / 180$  is identically equal to  $C_{ijk}$  of (12).

The indicated summation is conveniently expressed by

$$\phi_j \, f_{ijk} = \alpha_i \quad i, j, k = 1, 2, 3 \quad (19)$$



which is immediately recognized as  $\gamma^e_{\alpha_{ik}} = C^*_{ik}$ , the desired 3x3  
 element matrix. In this form it is combined into the global  $[C^*_{N \times N}]$ ,  
 and the linearization has been accomplished. As will be shown later,  
 it may be convenient to let  $\phi = \psi_{t-\Delta t}$  when for small  $\Delta t$ ,  $\phi =$   
 $\psi_{t-\Delta t} \approx \psi_t$ . This approximation is useful for predictor-corrector  
 equation solvers in order to allow construction of  $[C^*_{N \times N}]$  only once  
 for each successful time step.



### III. SOLUTION PROCEDURE WITH THE LINEARIZED FORM

For clarity, throughout the remainder of this thesis the following naming convention will be used:  $N^3$  denotes the nonlinearized direct treatment by DVOGER as solved in Ref. 3. "Linearized approximate" indicates the linearized technique based on element level multiplication of  $C$  by  $\psi_{t-\Delta t}$ . The "linearized exact" method, however, is the element level multiplication of  $C$  by the trial  $\psi_t$ . The compact, or  $N \times p$  formulation may be used with any of the linearized methods in addition to the CRANKO treatment discussed later.

#### A. GENERAL TREATMENT OF THE MATRIX EQUATION

Solution of the matrix formulation of Eq. (8) involves clearing the LHS to provide a discrete relation for each  $\dot{\psi}_k$ . This is accomplished either by iteration, which yields an approximation, or by matrix inversion, which has the advantage of giving the individual  $\dot{\psi}_k$  explicitly. In both the nonlinearized  $N^3$  and the linearized  $N \times N$  forms, Eq. (8) is reduced with multiplication by  $[A]^{-1}$ ; thus

$$[A]^{-1}[A]\{\dot{\psi}\} = [A]^{-1}[B]\{\psi\} + [A]^{-1}[C\phi]\{\psi\} \quad (20)$$

in the linearized  $\psi^2 = \phi\psi$  case or as

$$[A]^{-1}[A]\{\dot{\psi}\} = [A]^{-1}[B]\{\psi\} + [A]^{-1}[C]\{\psi^2\} \quad (21)$$

for the direct, nonlinear, formulation. This results in

$$\{\dot{\psi}\} = [AB]\{\psi\} + [A^{-1}][C^*]\{\psi\} \quad (22)$$



where

$$[C^*] = [C\phi] \quad \text{or} \quad [C]\{\psi\} \quad (23)$$

as appropriate. The point here is that the  $[AB]$  and the  $[A^{-1}]$  are formed once and for all based on the time independent properties and geometry of the problem, whereas the  $[C\phi]$  must be computed for each time  $t$ . Although the construction of the  $[C]$  is required only once in the direct ( $N^3$ ) method, it must be additionally multiplied by  $\{\psi^2\}$  at each time step.

#### B. PRACTICAL CONSIDERATIONS WITH THE LINEARIZED TECHNIQUE

Relating Eq. (22) to the demands of the subroutine DVOGER, which involves the evaluation of  $\dot{\psi}$  for every trial value of  $\psi$ , it is apparent that the construction of  $[A]^{-1}[C\psi]\{\psi\}$  for each of these trials could be a time consuming process, and hence a serious drawback of the linearized treatment. Noting, however, that for small  $\Delta t$ ,  $\psi_t \approx \psi_{t-\Delta t}$ , the approximation  $[C\psi] \approx [C\phi]$ , where  $\phi = \psi_{t-\Delta t}$ , overcomes this disadvantage by allowing the construction of  $[A]^{-1}[C\phi]$  only once for a given time  $t$ , regardless of how many trial solutions are attempted by DVOGER. Equation (22) can now be further simplified since  $[A]^{-1}[C\phi]$  is readily combined with  $[AB]$  to reformulate the problem as

$$\{\dot{\psi}\} = [C^{**}]\{\psi\} . \quad (24)$$

Thus, although the linearized form can be handled exactly, i.e., with  $\phi = \psi$ , it may be more expeditious to establish the  $[C^{**}]$  only once per valid time point, (that is, where the convergence criterion of the equation solution has been satisfied). In this case, the linearized treatment is then an approximate technique in that  $\phi = \psi_{t-\Delta t} \neq \psi_t$ .





### C. DIFFICULTIES WITH THE APPROXIMATION

The simplification expressed as Eq. (24) works extremely well when used with DVOGER as long as large changes in  $\psi$  are incurred by small  $\Delta t$  (the so-called "stiff" system). When the solution approaches steady state, however,  $\dot{\psi}$  approaches zero faster than  $(\psi_t - \psi_{t-\Delta t})/\Delta t$  does, and this causes a second kind of perturbation wherein the system becomes increasingly sensitive to small changes in  $\psi_t - \psi_{t-\Delta t}$ . Apparently DVOGER requires very accurate  $\dot{\psi}$  information. The  $\dot{\psi}(t)$  provided from  $\psi(t) \times \psi(t-\Delta t)$  does not match the DVOGER prediction of  $\dot{\psi}(t)$ , which is based on the assumption that  $\dot{\psi}(t)$  was generated by  $\psi(t)^2$ . This discrepancy in the neighborhood of  $\dot{\psi} \approx 0$  causes DVOGER to be unable to recognize the steady state solution. Prepared especially for stiff systems, the equation-solver expects, instead, a large scale change in  $\psi$ . The time step is therefore reduced accordingly, which means that very slow progress is made in the steady-state region which is otherwise handled quite rapidly under the exact methods. The implication here is that, in those cases which reach terminal flux values rather early in problem time, the exact formulation of the linearized treatment is to be preferred, even though the  $[C^{**}]$  must be computed for each attempted  $\psi(t)$ . This aspect has been analyzed for the test problems considered, with the results given in Table II.



#### IV. THE NxP COMPACTING SCHEME

##### A. ESTABLISHED COMPACTING METHOD

The problem under consideration here, being of the form  $A_{ij} \dot{\psi}_j = B_i(\psi_1 \dots \psi_N)$ , is formally handled by multiplying through by  $[A]^{-1}$ , discussed earlier. For large matrices, the inversion process, and indeed all matrix operations, become extremely costly in terms of machine time and storage. The fact that the matrices may be banded or symmetric does offer significant economy when the matrix is not inverted. For example, a symmetric  $N \times N$  matrix may be stored as  $N \times N/2$ . For matrices resulting from finite element formulation, the numbering schedule determines a band width to be used with conventional compacting schemes. In the finite element case, the difference is compared for each node in the mesh. The largest of these differences plus one equals the minimum band width allowed. For example, node 7 in Figure 1 has a maximum difference of  $13 - 2 = 11$ , whereas node 22 has the largest difference ( $37 - 10 = 27$ ) for the system. Thus the resultant minimum band width would be 28. As the mesh increases in nodal points, the minimum band width must also increase in response, no matter how adroitly the numbers are assigned. Regardless of how sparse a matrix may be, these schemes all ensure that the "compactness" grows commensurately. In addition, the calculation of an inverse becomes ever more unpleasant, especially since this operation on a sparse matrix generally results in a dense, nonsymmetric inverse.



## B. BASIS FOR THE COMPACTING SCHEME

In solving a matrix equation  $A_{ij}\dot{\psi}_j = B_i(\psi_1, \dots, \psi_N)$  it is not necessary to form the inverse, but rather, multiply out the left hand side and rearrange to obtain a set of equations:

$$\dot{\psi}_i = (B_i - A_{ij}\dot{\psi}_j)/A_{ii} \quad (25)$$

where  $i, j, k = 1, \dots, N$ . This procedure is especially suitable for sparse systems since the number of non-zero  $A_{ij}$  entities will be small. Further, if these non-zero  $A_{ij}$  can be located and tagged with an identifier, then they may be stored together in a dense array, thereby replacing the standard  $N \times N$  size matrix by an equivalent array of size  $N \times p$ , where  $p$  is the maximum number of non-zero entries on any one row.

This type of compacting is especially well suited to matrices resulting from finite element formulation, with  $p$  determined by the choice of the finite element and the discretized model. In the case under study, a linear triangular element is used such that, when assembled on the global level, it forms, for the purpose of illustration, a network of interlocking hexagonal polygons, each having an apex located directly over its parent central node. Such a pattern is sketched as Figure 2. Note that the sides are indeed the contributing "neighbor" nodes. Boundary nodes will, of course, be missing any "would be" contributors sought in the boundary exterior. In a regularly drawn system, as in Figure 2, an interior node will have six neighbors plus itself for a total of seven contributors. That is to say, that no matter how large the system of mesh points, a relation describing the effect of the system on any given point would contain a maximum of seven non-zero values.



Geometric considerations or a desire to examine some area of the mesh more closely may result in systems of irregular discretization wherein the maximum number of entries may be greater than seven. This analysis may be extended to consider other finite element shapes in a similar manner.

#### C. CONSTRUCTION OF THE $N_{xp}$ ARRAY

To utilize this attribute of finite elements to produce the dense  $N_{xp}$  system, it is necessary only to construct a table of nodal points listing their contributors. This table, or connectivity array, is then used as an index to locate the values of the various quantities associated with a particular node. All the standard matrix operations can be performed on these compacted arrays, but the construction of a matrix inverse has no usefulness since a  $N \times N$  system is then reconstituted from an  $N_{xp}$  array.

As an example, construction of the connectivity vector associated with node 18 in Figure 1 is

$$[18 \ 17 \ 27 \ 26 \ 19 \ 14 \ 13 \ 12]$$

and the vector associated with node 6 is

$$[6 \ 11 \ 7 \ 21 \ 0 \ 0 \ 0 \ 0].$$

The contributors may be entered in any order except that, for computational ease, the central, or "parent" node is the first element. These vectors are collected into the node neighbor connectivity array of size  $N_{xp}$ , or, for Figure 1,  $38 \times 8$ .





To multiply a  $N \times p$  array of  $a_{ij}$  elements by a vector of  $b_j$  elements, a search of the connectivity array is performed on the "i"th row to locate a match for the  $b_j$ . When the match is found, the values represented by  $a_{ij}$  and  $b_j$  are multiplied together resulting in a new  $C_i$  vector. If the system is correctly formed and compacted, there always will be a match.

The compacting scheme is bound by the geometry of the problem, and as such, has meaning only as it pertains to that problem. For example, operations with two equi-sized  $N \times p$  arrays of different connectivity relations cannot be performed. This is in contrast to regular matrix operations where the restriction is only to size and not to origin. A short computer program to demonstrate the operation of this  $N \times p$  scheme is given, with results, in Appendix C.



## V. THE CRANK-NICOLSON FORMULATION

Recognizing the very sparse nature of the system of differential equations formulated by this problem and, if the matrix inversion process is to be avoided, the necessity of iteration, a straightforward attack based on the definition of the derivative (finite difference) appears as a feasible alternative to the DVOGER technique. One such approach was presented by J. Crank and P. Nicolson in 1947 [Ref. 7] and has been discussed in many works throughout the ensuing years. This method has been shown to be unconditionally stable. An adaptation of this method is given here as follows:

In the general sense, a matrix formulation of a set of linear differential equations, can be represented as

$$A_{ij} \dot{\psi}_j = C_{ij} \psi_j + F_j(t) \quad i, j = 1, \dots, N \quad (26)$$

where  $\psi$  is a function of time. For the case of an initial disturbance as the forcing function,  $F(t) = 0$  for  $t > 0$ . Focusing attention for the moment on a particular equation and expressing [16] in terms of the definition of the derivative

$$A \left[ \frac{\psi_t - \psi_{t-\Delta t}}{\Delta t} \right] = C \left[ \frac{\psi_t + \psi_{t-\Delta t}}{2} \right] \quad (27)$$

which is

$$[A - 1/2 \cdot \Delta t \cdot C] \psi_t = [A + (1/2 \cdot \Delta t \cdot C)] \psi_{t-\Delta t} \quad (28)$$

$$[(2/\Delta t \cdot A) - C] \psi_t = [(2/\Delta t \cdot A) + C] \psi_{t-\Delta t} \quad (29)$$



letting  $D = (2/\Delta t \cdot A) - C$ ,  $E = (2/\Delta t \cdot A) + C$ ,  $\phi = \psi_{t-\Delta t}$  then, returning to the system form the result is

$$D_{ij}\psi_j = E_{ij}\phi_j = E_i^* \quad i, j = 1, \dots, N. \quad (30)$$

This constitutes a set of linear simultaneous equations which may be solved by any convenient method. Chosen here is Gauss-Seidel iteration [Ref. 8] which is well suited for the problem at hand since, the sum of  $D_{ij}\psi_j$  and  $E_{ij}\phi_j$  will contain very few non-zero terms, thereby incurring relatively small roundoff errors regardless of system size.

This procedure, which also uses the Nxp compacting, has been written for this thesis as the computer code CRANKO. After receiving the matrix information and control parameters from the calling program, CRANKO first builds the C matrix of Eq. (26) for the initial  $\phi$  value. Then it selects a trial time interval and, forming the equation set (30) based on this  $\Delta t$ , attempts to find a solution satisfying a specified convergence criterion based on a relative error. If no satisfactory solution set is found after a number of iterations, a smaller  $\Delta t$  is selected; and, after recomputation of Eq. (30), another attempt at convergence is made. The successful solutions then replace the previous  $\psi$  thereby forming the starting point for another cycle, beginning with a new computation of the C matrix. The choice of  $\Delta t$  is the controlling factor. The scheme employed counts the number of iterations required for a solution. When an increasing number of iterations indicate greater difficulty, the interval  $\Delta t$  is decreased; whereas, if fewer iterations are required, the time step may be relaxed and made larger. Experience with the CRANKO routine establishes this method as an extremely fast technique for solving this type of problem.



For those readers unfamiliar with the Gauss-Seidel iteration scheme,  
a simple example using a 3x3 system is given:

$$D_{ij}\psi_i = E_j^* \quad \phi = \psi_{t-\Delta t}$$

$$\begin{bmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{bmatrix} \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \end{bmatrix} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}$$

first iteration:

$$d_{11}\psi_1 + d_{12}\phi_2 + d_{13}\phi_3 = e_1$$

$$\psi_1 = [e_1 - (d_{12}\phi_2 + d_{13}\phi_3)]/d_{11}$$

$$\psi_1^* = \psi_1$$

$$d_{21}\psi_1^* + d_{22}\psi_2 + d_{23}\phi_3 = e_2$$

$$\psi_2 = [e_2 - (d_{21}\psi_1^* + d_{23}\phi_3)]/d_{22}$$

$$\psi_2^* = \psi_2$$

$$d_{31}\psi_1^* + d_{32}\psi_2^* + d_{33}\psi_3 = e_3$$

$$\psi_3 = [e_3 - (d_{31}\psi_1^* + d_{32}\psi_2^*)]/d_{33}$$

for successive iterations,





$$d_{11}\psi_1 + d_{12}\psi_2^* + d_{13}\psi_3^* = e_1$$

$$\psi_1 = [e_1 - (d_{12}\psi_2^* + d_{13}\psi_3^*)]/d_{11}$$

$$\psi_1^{**} = \psi_1$$

$$d_{21}\psi_1^{**} + d_{22}\psi_2 + d_{23}\psi_3^* = e_2$$

$$\psi_2 = [e_2 - (d_{21}\psi_1^{**} + d_{23}\psi_3^*)]/d_{22}$$

$$\psi_2^{**} = \psi_2$$

$$d_{31}\psi_1^{**} + d_{32}\psi_2^{**} + d_{33}\psi_3 = e_3$$

$$\psi_3 = [e_3 - (d_{31}\psi_1^{**} + d_{32}\psi_2^{**})]/d_{33}$$

etc.



## VI. SUMMARY OF AVAILABLE METHODS

For clarity, the innovative computational tools introduced in this thesis are summarized. First is the linearizing of the governing equation Eq. (8) by the multiplication on the element level of  $C_{ijk}$  by  $\psi_j$ . From this development, two options appear feasible: either the exact computation of  $[C^*]$  for each trial  $\psi$ , or the approximation of  $\psi_t \approx \psi_{t-\Delta t}$  to allow the construction of  $[C^*]$  only once during the search for a new  $\psi_t$ . This approach uses, as does the "direct," or nonlinear treatment involving  $[C]$ , the inverting of  $[A]$  to clear the LHS as shown in Eq. (22).

The second innovation is the reduction of the  $N \times N$  matrix to a smaller  $N \times p$  compact form based, not upon the mathematics of the problem, but upon the geometry of the finite element model. The same solution methods may be used, with the only difference being that the inverting of  $[A]$  is replaced by iteration. This iteration can only be as accurate as the imposed error criterion. For large systems, however, where the  $[A]^{-1}$  produces a full  $N \times N$  matrix on the RHS, this iteration is especially beneficial.

The last contribution was the development of a new equation-solver based on a different theory than the DVOGER routine. The new code deals with the differential equations directly and employs the  $N \times p$  compacting.



## VII. TEST PROBLEMS AND RESULTS

### A. THE PHYSICAL MODEL

A cylindrical nuclear reactor consisting of a core region and blanket having overall a height of 220 cm and a diameter of 180 cm was modeled for the test problems. As shown in Figure 1, a radial slice from this reactor was discretized by a finite element of 54 elements and 38 nodal points. After a detailed analysis of this mesh was completed, preliminary investigation was begun using a model of 220 elements (Figure 3).

### B. COMPUTER PROCESSING CONSIDERATIONS

All programs were written in the FORTRAN IV language and processed on the IBM 360/67 computer using the FORTRAN 'H' compiler. The results presented in terms of CPU storage requirements and processing time ought to be indicative, in a relative way, of those obtained from other machines. Single precision (seven significant digits) was used throughout the investigation, with the results later verified using double precision (fifteen significant digits). The observed relative discrepancy was less than .01%. It is recognized, however, that for larger systems a more significant difference may result.

### C. PROBLEM ANALYSIS

The three techniques discussed have been incorporated into computer codes and have been tested on those problems described in Reference 3, namely, the dynamic reactor response to 1) a disturbance at the center, 2) a uniform disturbance throughout the core, and 3) a disturbance occurring at a skew point in the core ( $R = 40$  cm,  $Z = 0$  cm). The direct, or  $N^3$



matrix (an  $N \times (N^2)$  array) equation was solved in Reference 3, and those results are used in this thesis as a standard for comparison. The computer codes developed here have been constructed within the framework and parameters of the original  $N^3$  program, thereby allowing any discrepancy to be attributed to the technique employed. For each technique and disturbance, the flux at three sample points was recorded throughout the time history of the solutions and written onto a computer storage device. A separate program was written to process this information and produce the correlations in tabular and graphic form. The graphic results are presented in Figures 4 - 21 for the center point ( $R = 0$  cm,  $Z = 0$  cm), a core point ( $R = 40$  cm,  $Z = 40$  cm), and a reflector point ( $R = 75$  cm,  $Z = 80$  cm). Table II lists the specific programs employed arranged by disturbance type and gives parameters such as storage requirements and computer processing time.

#### 1. Solutions with the CRANKO Subroutine

All techniques appear to give acceptable accuracy, however there is a startling difference in processing time. The CRANKO subroutine was able to accomplish in seconds the work which required minutes for the DVOGER subroutine (See Table II). The steady state solution given by CRANKO was in every case higher than that obtained by DVOGER. The implication here is not clear since there is no analytical solution available to represent the exact answer. In general, the transient solutions agreed well, except for the oscillation of the CRANKO solution in the early stages, most clearly depicted in the extreme case of Figures 8 and 9. This fluctuation is most likely the result of inadequate control of the CRANKO time step selection where apparently an interval has been chosen which is too large to maintain good accuracy. It is noteworthy,





however, that the technique is so stable that even after large deviation it is able to recognize the error, correct itself, and return to the solution curve given by the DVOGER routine.

## 2. Solutions with the DVOGER Subroutine

The linearized technique using premultiplication by the previous time step flux ( $\psi_{t-\Delta t}$ ) conformed to theoretical expectations. As the steady state solution was approached, this method proceeded with increasing difficulty. Hence much more processing time to advance through the steady state was required.

The direct  $N^3$  method, on the other hand, has no difficulty in recognizing the steady state and marches forward with large time increments until problem termination. A very successful effort was made to use the exact treatment with the linearized technique. As expected, more processing time was consumed in getting to the steady state due to the requirement to recompute the  $[C^*\psi]$  for each trial  $\psi$ , but once the steady state was approached, the procedure progressed rapidly as in the direct treatment. Comparing the direct with the linearized exact method, theory developed in this thesis predicts identical results with twice the computing time required by the linearization. As shown in Table II, this is indeed the case. The largest relative error observed between the two methods was .06%, which is attributed to roundoff stemming from the doubling of the computational requirements. When the greatest accuracy is required, the exact treatment with linearization may therefore prove the most useful, especially if the problem rapidly reaches steady state. This discussion of the linearized technique is applicable both to the  $N \times N$  method or to the  $N \times p$  compact method. However since the  $N \times p$  method requires iteration, more processing time and less accuracy might be expected.



#### D. EVALUATION OF ERROR

The error analysis for these procedures must remain subjective as there is no "correct" result. Several sources of error are present and deserve comment. Tests were made with both  $EPS = .1$  and  $EPS = .01$ , the error criterion for the DVOGER routine, and little difference was noted for the large increase in processing time. [EPS is defined by DVOGER as  $\sum_{i=1}^N \frac{ERROR(i)}{Y_{MAX}}$ ]. The convergence criterion [relative deviation of any  $\psi_i$  between successive iterations] for the iterative solutions in CRANKO was varied by tenths from .01 to .001 with the result of producing higher steady state values of only a few percent as the criterion was made more stringent. Processing time for this variation increased only a few seconds. Double percision trials for the linearized DVOGER method and the compact CRANKO were made with neglibible difference.

The grid of only 54 elements theoretically introduces the largest source of error from the "true" solution. With the techniques developed in this thesis, a finer grid size of 220 elements has been implemented into the CRANKO and the linearized codes. Because the linearized code (DVOGER) requires about 566K core size, extensive testing with this technique is not anticipated, but the CRANKO routine which requires only 188K is currently being investigated. As the results of this investigation become clear, a better understanding of the error induced by each of the various approaches to this problem would be obtained.



## VIII. CONCLUSION

It appears that nonlinear differential equations resulting from a finite element formulation (or for that matter finite differencing) may be successfully reduced using the linearizing technique developed in this thesis. The compacting scheme for matrix equations effects gross savings for storage of large systems (Table I) and becomes even more attractive when the inverse of a matrix is either not desired or required. The Crank-Nicolson formulation coupled with Gauss-Seidel iteration as expressed in the computer code CRANKO has demonstrated itself to be a viable, extremely fast technique for the solution of differential equations. Further improvements to the CRANKO routine, specifically in regard to better control of the time step selection, may improve upon the attractiveness of this approach.

Development is warranted in several areas. The computation of the Jacobian using iteration must be formulated in order to use the DVOGER subroutine with the compact system. DVOGER itself might usefully be reviewed to reduce its size requirements also. Investigation of the problems using the fine mesh size of 220 elements is continuing. This should yield more exact results and offer a better standard for comparison.



APPENDIX A

TABLES





TABLE I  
IBM 360/67 CORE REQUIREMENTS  
in K Bytes

<u>SYSTEM</u>	<u>NO. NODES</u>	<u>OVERHEAD</u>	<u>MATRIX STORAGE</u>	<u>TOTAL</u>
$N \times N^2$	38	100	225	325
$N \times N$	38	100	35	135
$N \times p$	38	100	12	112
$N \times N$	132	150	418	568
$N \times p$	132	150	38	188
$N \times p$	400	290	112	402 †

†calculated estimate



TABLE II  
COMPARISON OF METHODS  
38 NODAL PT. MODEL

<u>METHOD</u>	<u>CORE (k)</u>	<u>PROCESSING TIME (MIN)</u>			<u>STEADY-STATE ERROR (%)</u>
		<u>CENTER</u>	<u>UNIFORM</u>	<u>SKEW</u>	
DVOGER $N^3$ .01	330	>	5.33	6.27	-
DVOGER $N^3$ .1	330	5.32	3.28	4.57	<.01
DVOGER NxN exact	130	10.22	6.07	8.10	.06
DVOGER NxN approx	130	10.73	>15.	>10.	.16
CRANKO	110	.62	.20	.22	5.
CRANKO DP	158	.77	.23	.25	5.

Notes:

1. Processing time is that time required by the program to reach one second in problem time. Steady state was obtained for each run.
2. .01 & .1 pertain to DVOGER error criterion EPS. All NxN trials used EPS = .1 CRANKO used iteration convergence tolerance of .0001.
3. Double precision (DP) trials showed a difference in the fifth significant digit when compared to the SP counterparts.
4. NxN approximate technique proceeds as rapidly as the  $N^3$  method until steady state is approached. This has been verified by experiments using various processing time cutoff points.



APPENDIX B

FIGURES



# REACTOR MODEL WITH 38 NODES

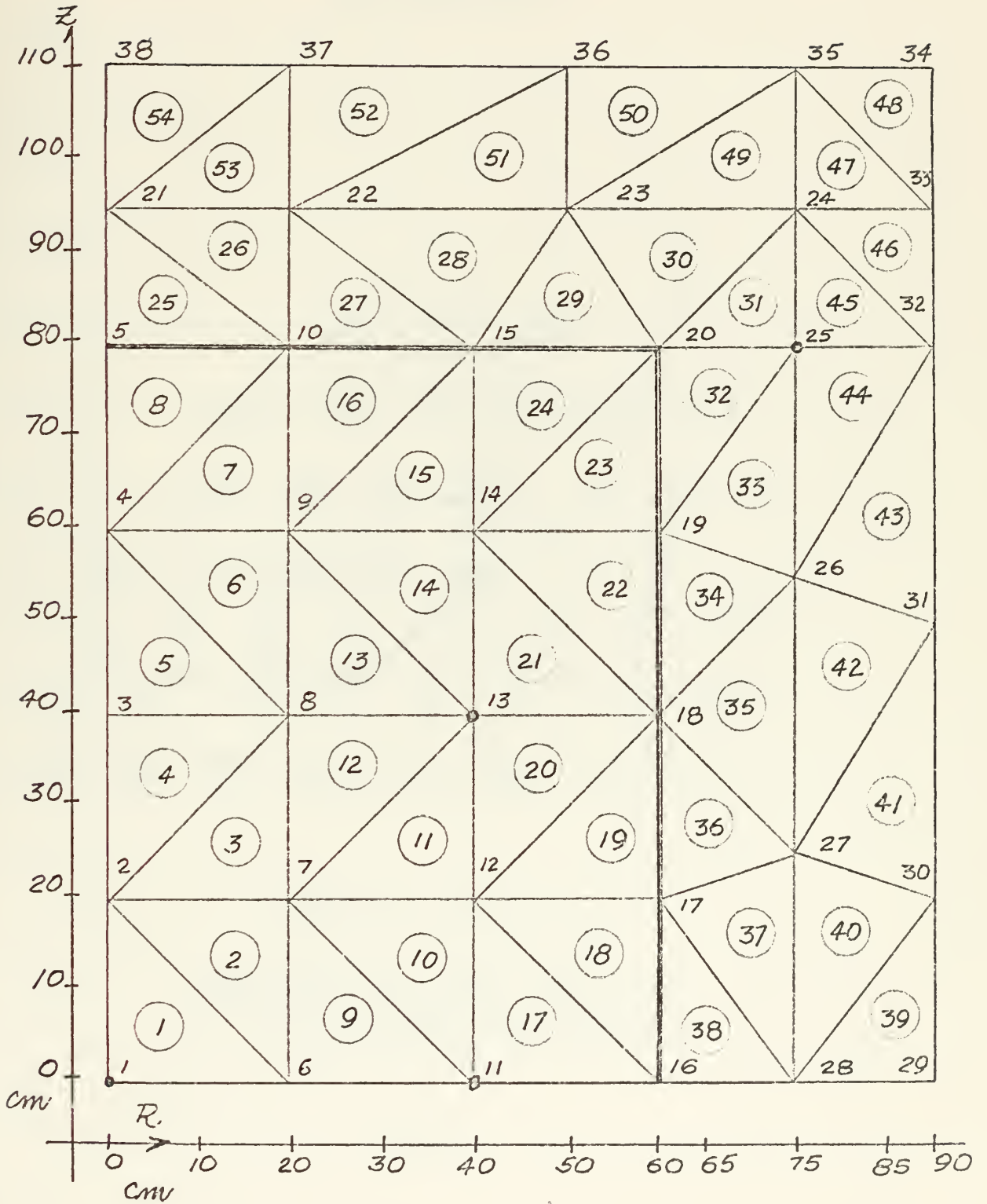
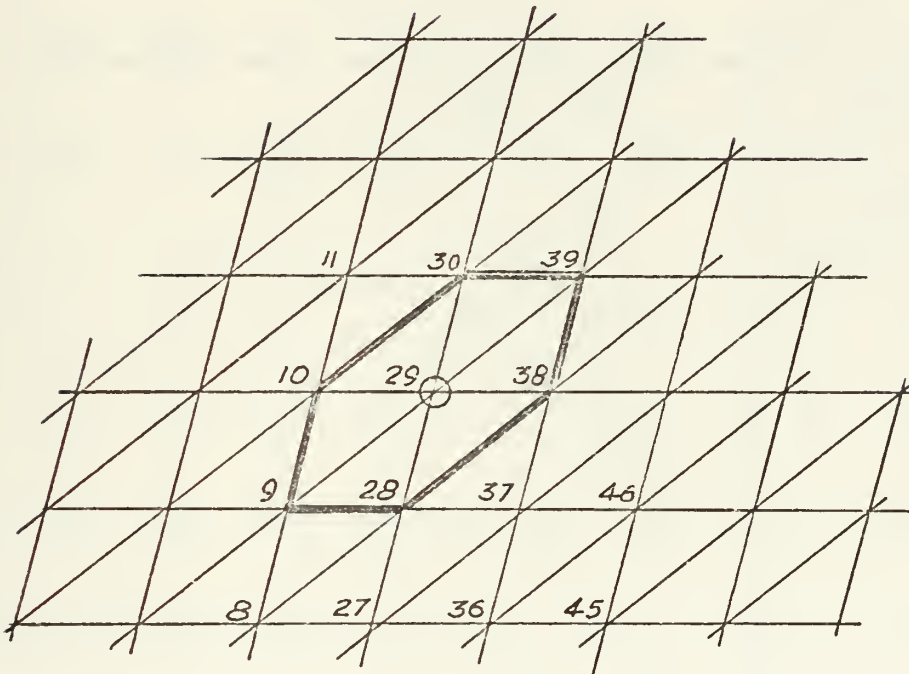


FIGURE 1





# NODAL CONNECTIVITY



CONNECTIVITY FOR NODE 29 IS

29 9 10 30 39 38 28

THESE NODES FORM CORNER POINTS OF ALL  
THE TRIANGULAR ELEMENTS WHICH CONTRIBUTE  
TO OR INFLUENCE POINT 29

FIGURE 2



## REACTOR MODEL WITH 132 NODES

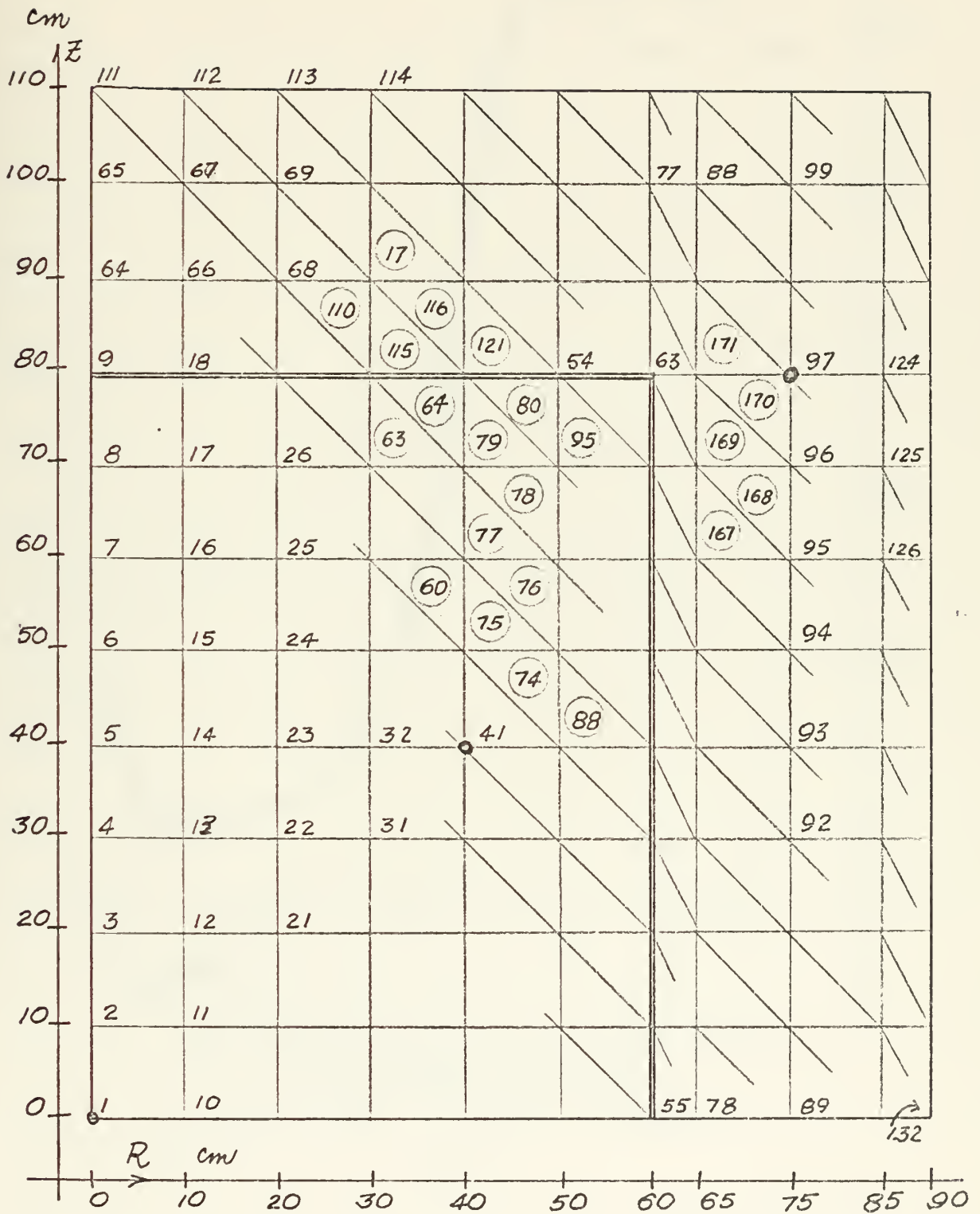
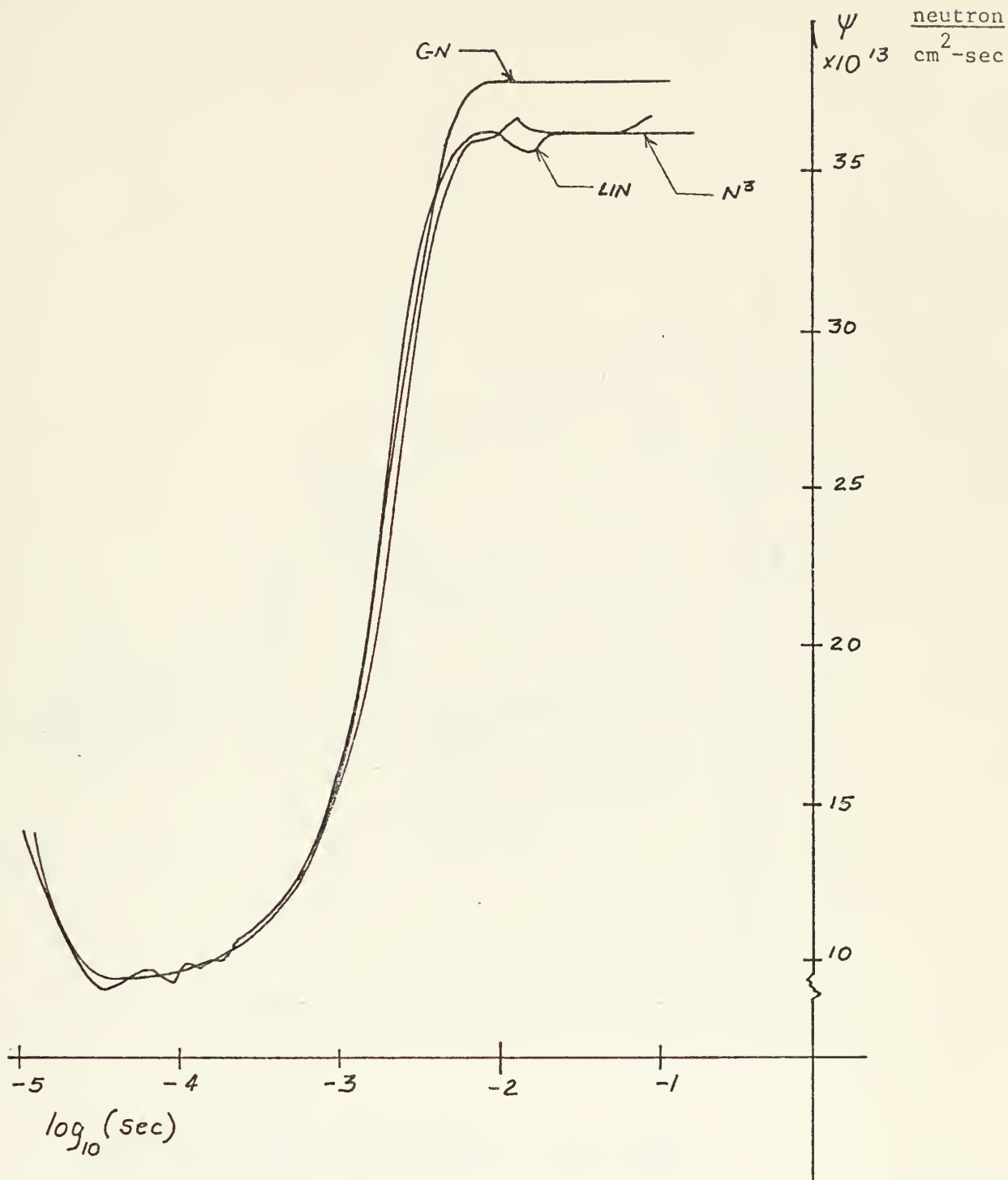


FIGURE 3

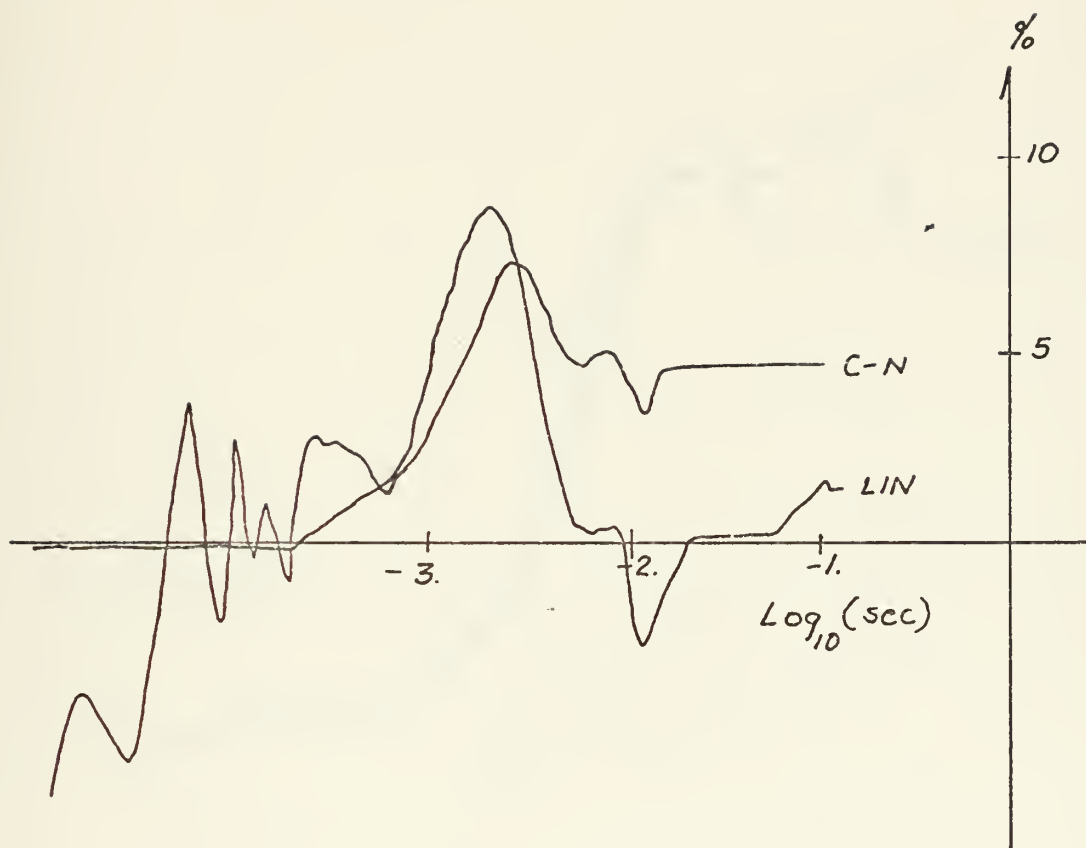




TIME DEPENDENT NEUTRON FLUX AT REACTOR CENTER  
FOR A CENTRAL DISTURBANCE  
PLOT OF CENTER DISTURBANCE - CENTER POINT

.. FIGURE 4 .



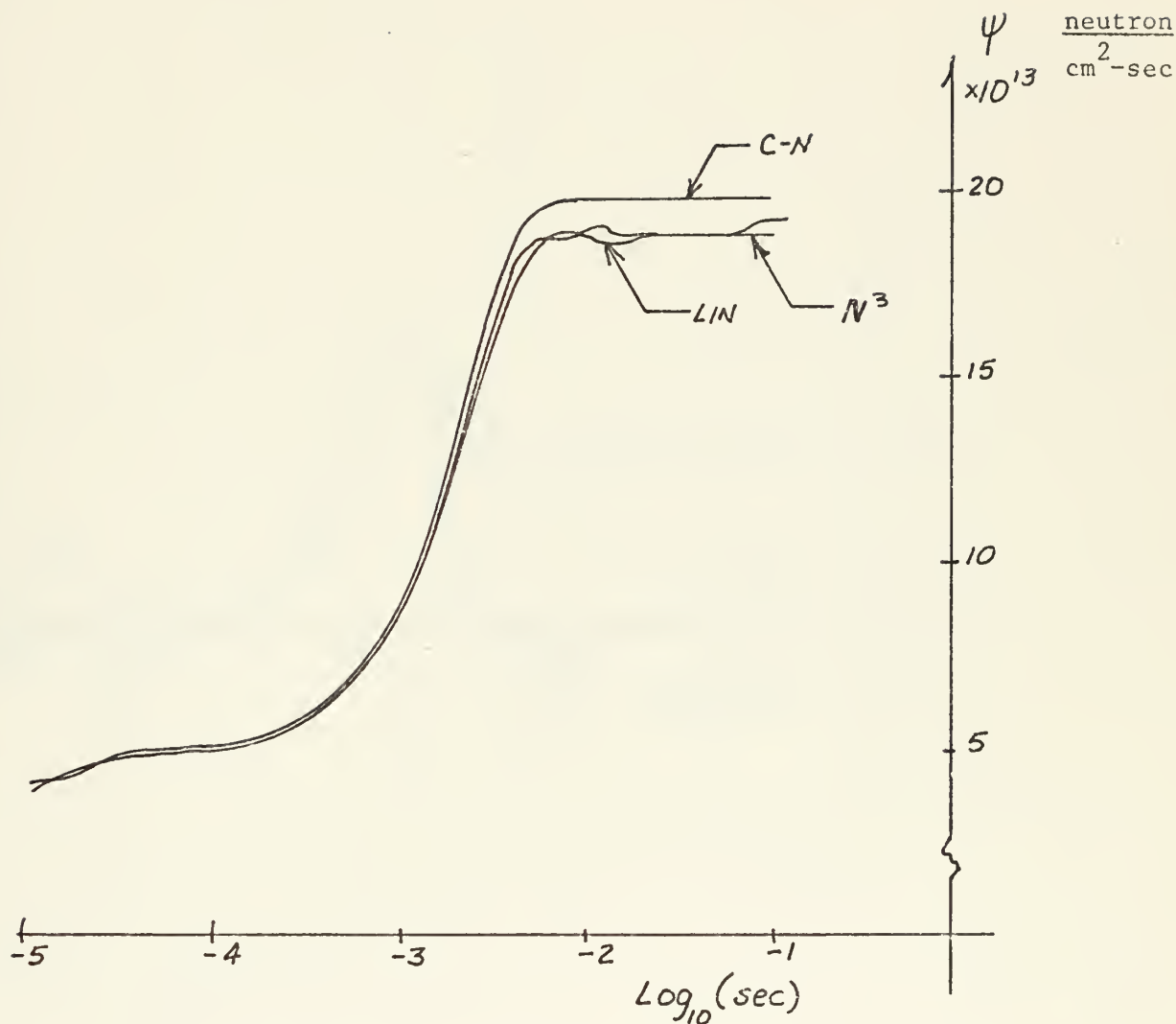


DEVIATION FROM  $N^3$  SOLUTION AT REACTOR CENTER  
COMPARISON - CENTER DISTURBANCE - CENTER POINT

FIGURE 5



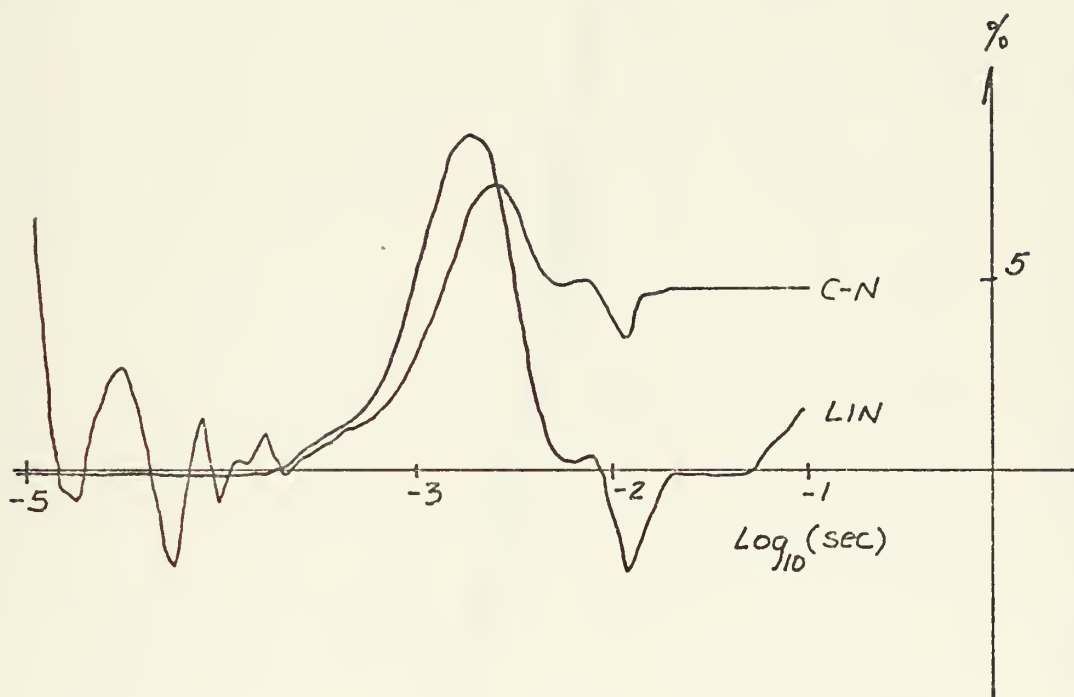




TIME DEPENDENT NEUTRON FLUX AT CORE POINT  $Z=40\text{cm}$ ,  $R=40\text{cm}$   
 FOR A CENTRAL DISTURBANCE  
 PLOT OF CENTER DISTURBANCE - CORE POINT

FIGURE 6





DEVIATION FROM  $N^3$  SOLUTION AT CORE POINT  $Z=40\text{cm}$ ,  $R=40\text{cm}$   
 COMPARISON - CENTER DISTURBANCE - CORE POINT

FIGURE 7



PLOT OF CENTER DISTURBANCE - REFLECTOR  
POINT

TIME DEPENDENT NEUTRON FLUX AT REFLECTOR  
POINT Z=80cm, R=75cm FOR A CENTRAL DISTURBANCE

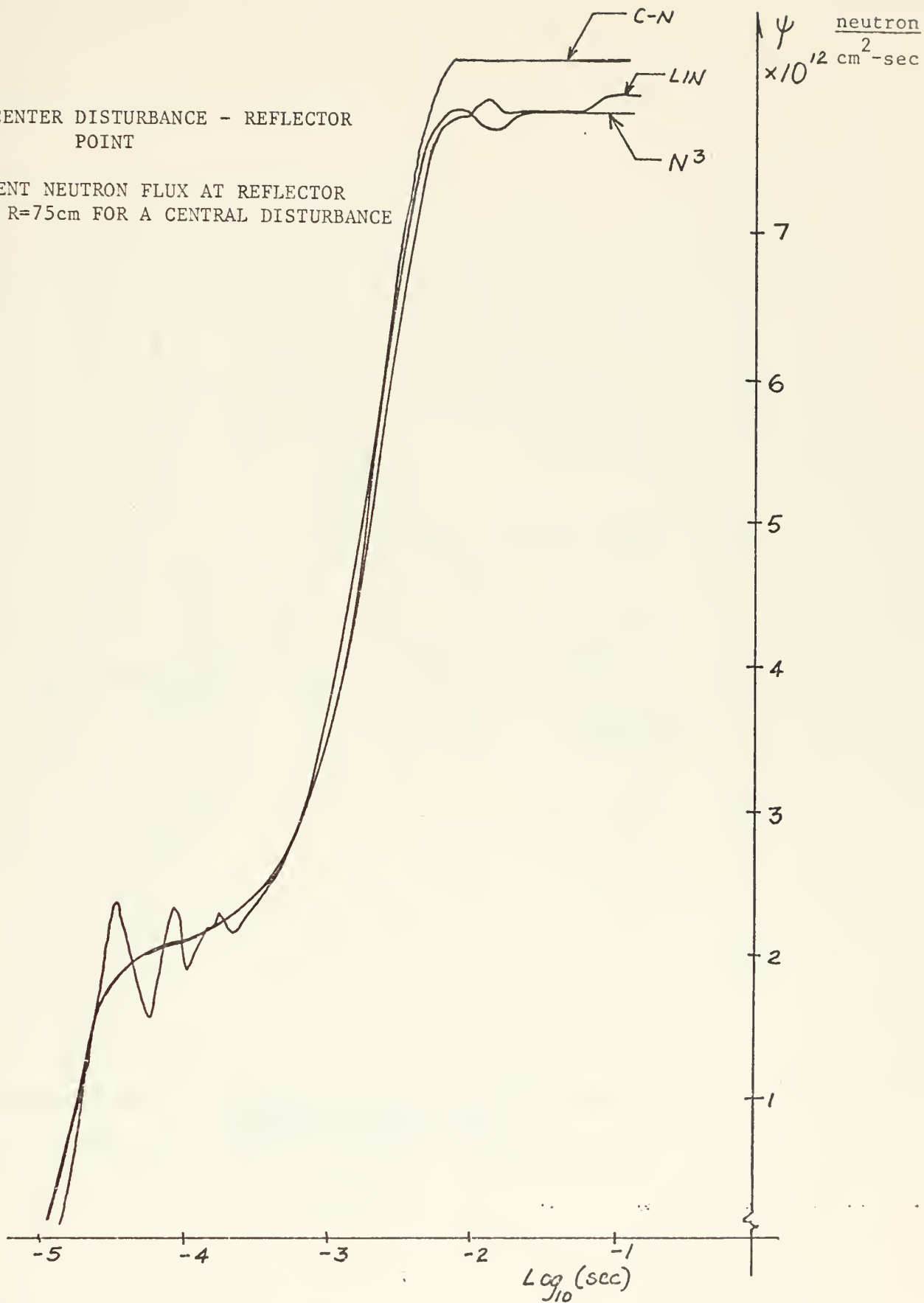
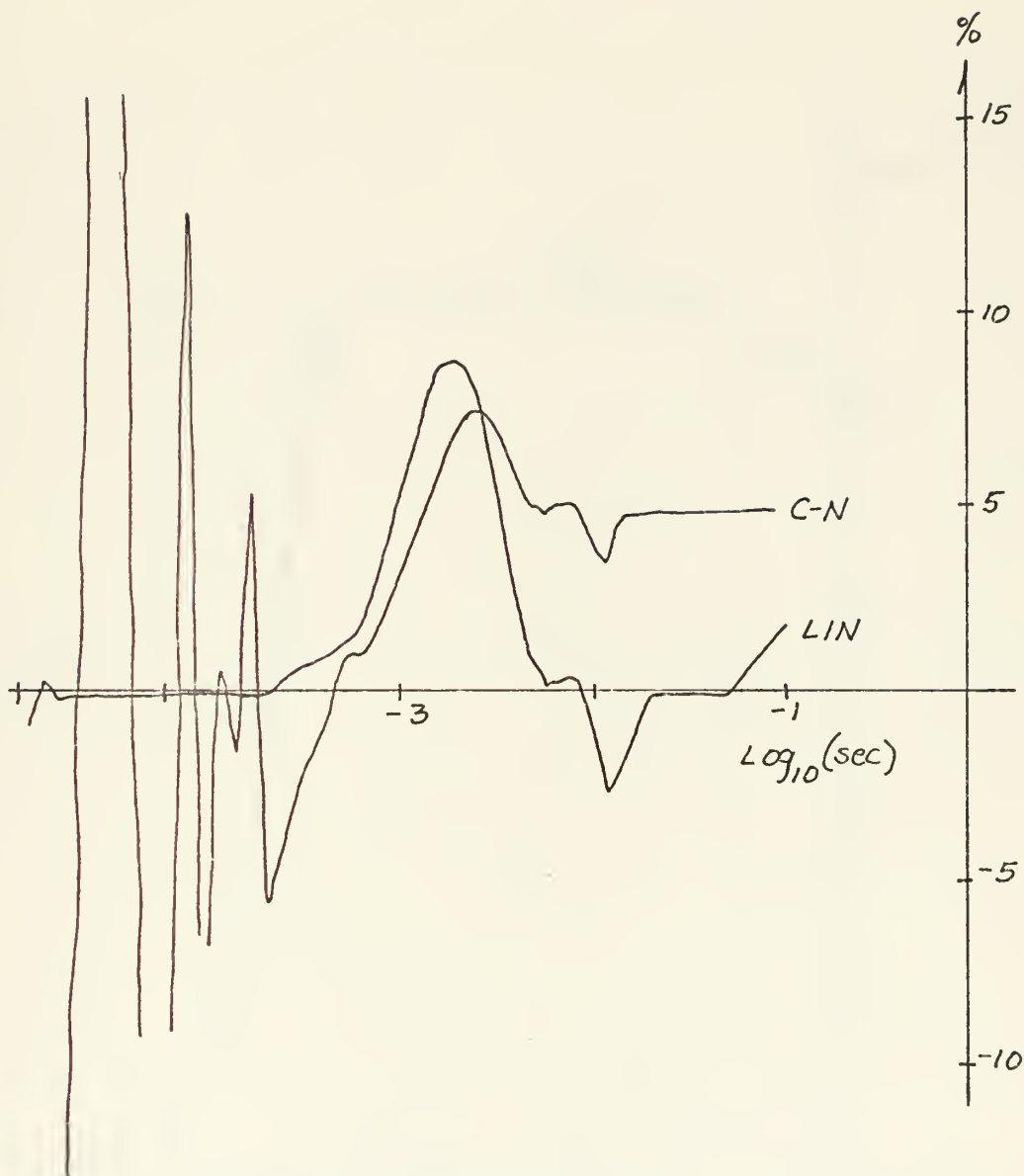


FIGURE 8



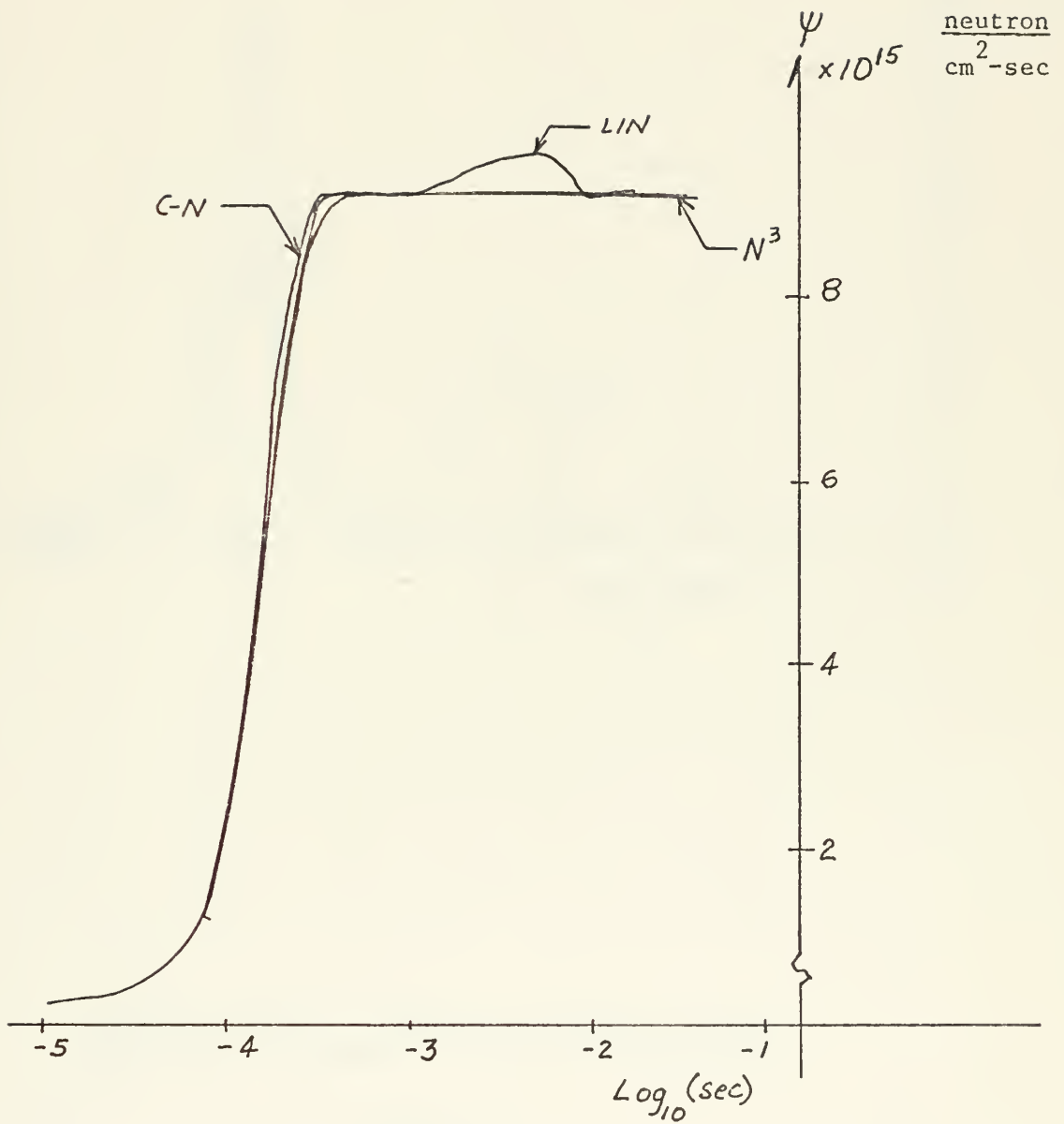


DEVIATION FROM  $N^3$  SOLUTION AT REFLECTOR POINT  $Z=80\text{cm}$ ,  $R=75\text{cm}$   
 AT REFLECTOR TEST POINT  
 COMPARISON - CENTER DISTURBANCE - REFLECTOR POINT

FIGURE 9



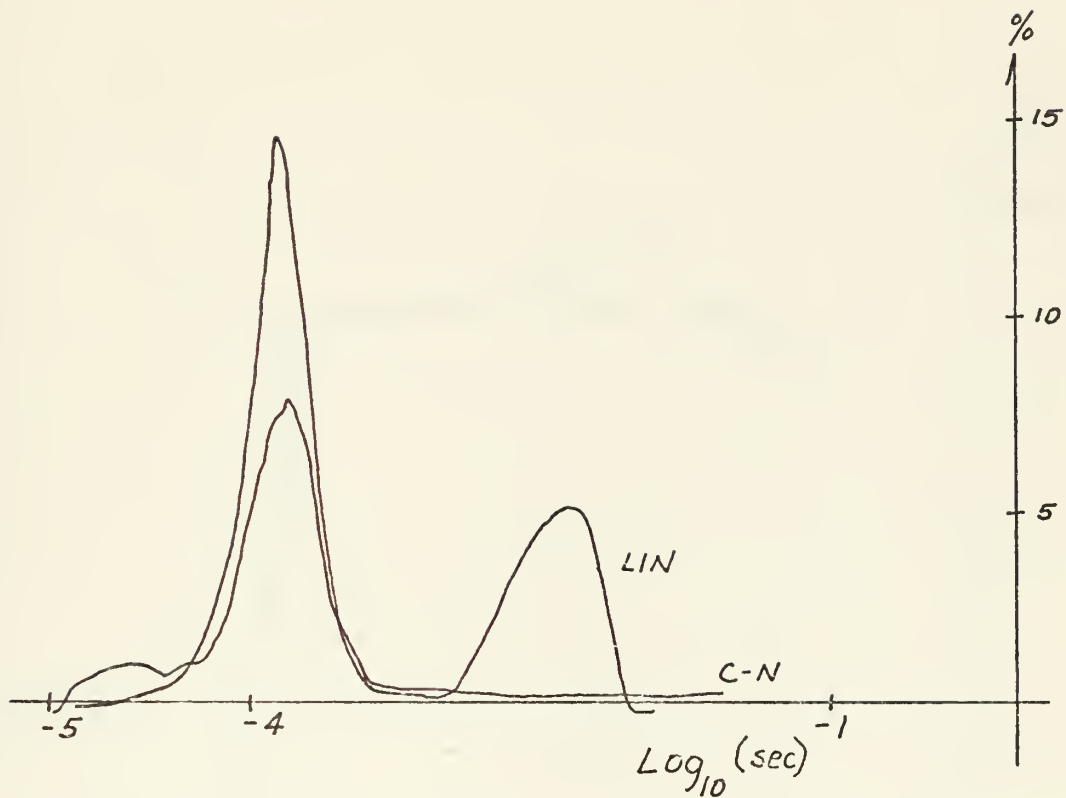




TIME DEPENDENT NEUTRON FLUX AT REACTOR CENTER  
FOR A UNIFORM DISTURBANCE  
PLOT OF UNIFORM DISTURBANCE - CENTER POINT

FIGURE 10

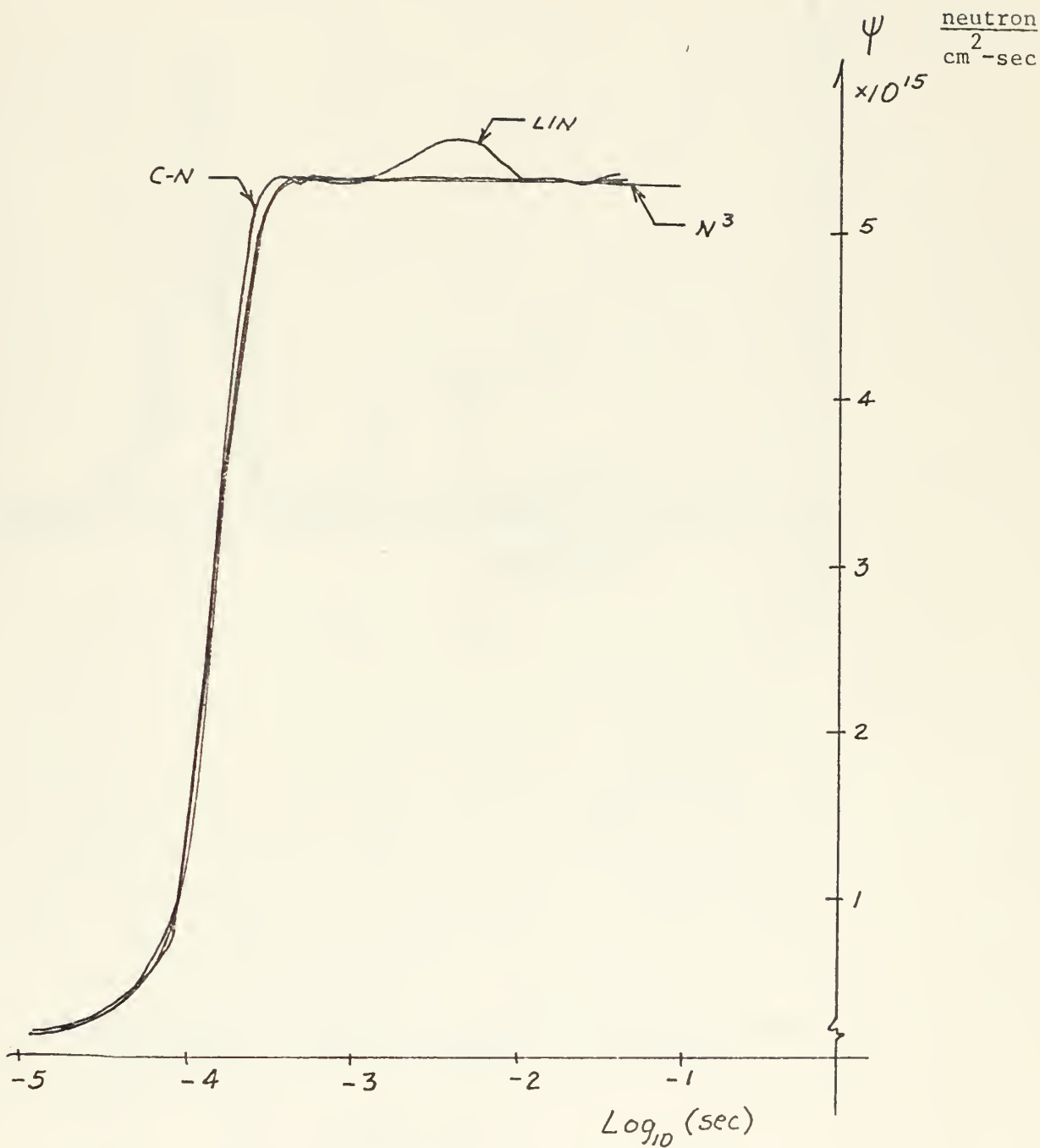




DEVIATION FROM  $N^3$  SOLUTION AT REACTOR CENTER  
COMPARISON - UNIFORM DISTURBANCE - CENTER POINT

FIGURE 11

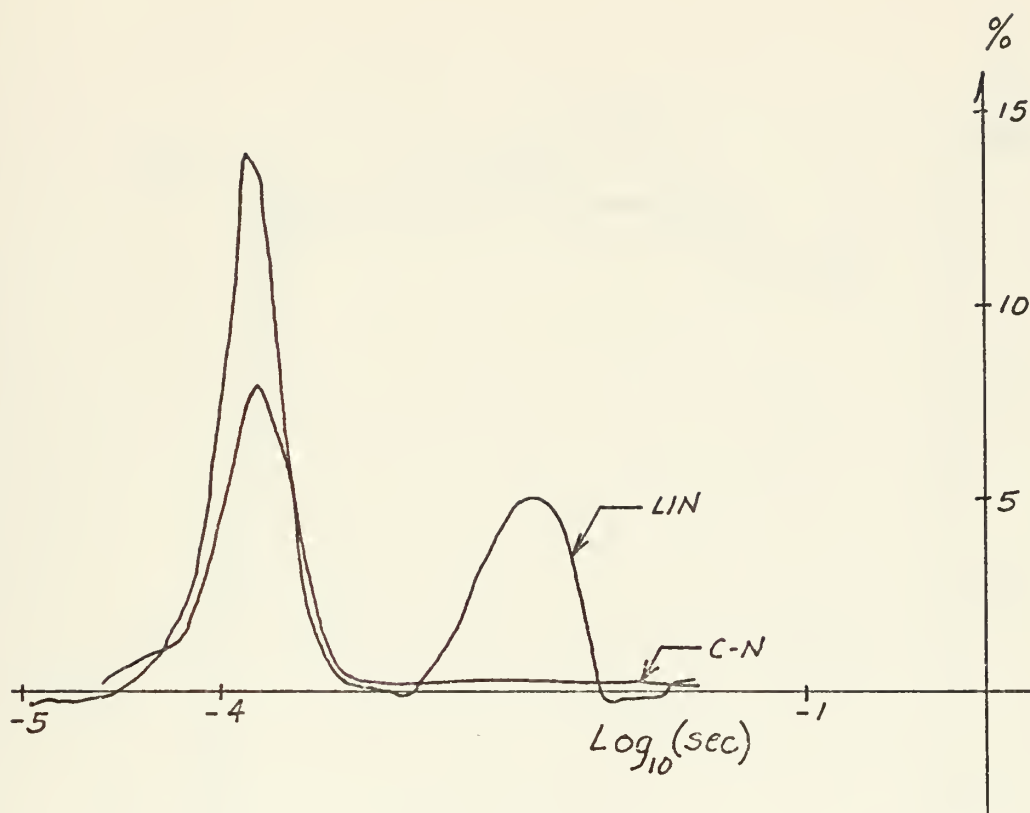




TIME DEPENDENT NEUTRON FLUX AT CORE POINT  $Z=40\text{cm}$ ,  $R=40\text{cm}$   
 FOR A UNIFORM DISTURBANCE  
 PLOT OF UNIFORM DISTURBANCE - CORE POINT

FIGURE 12



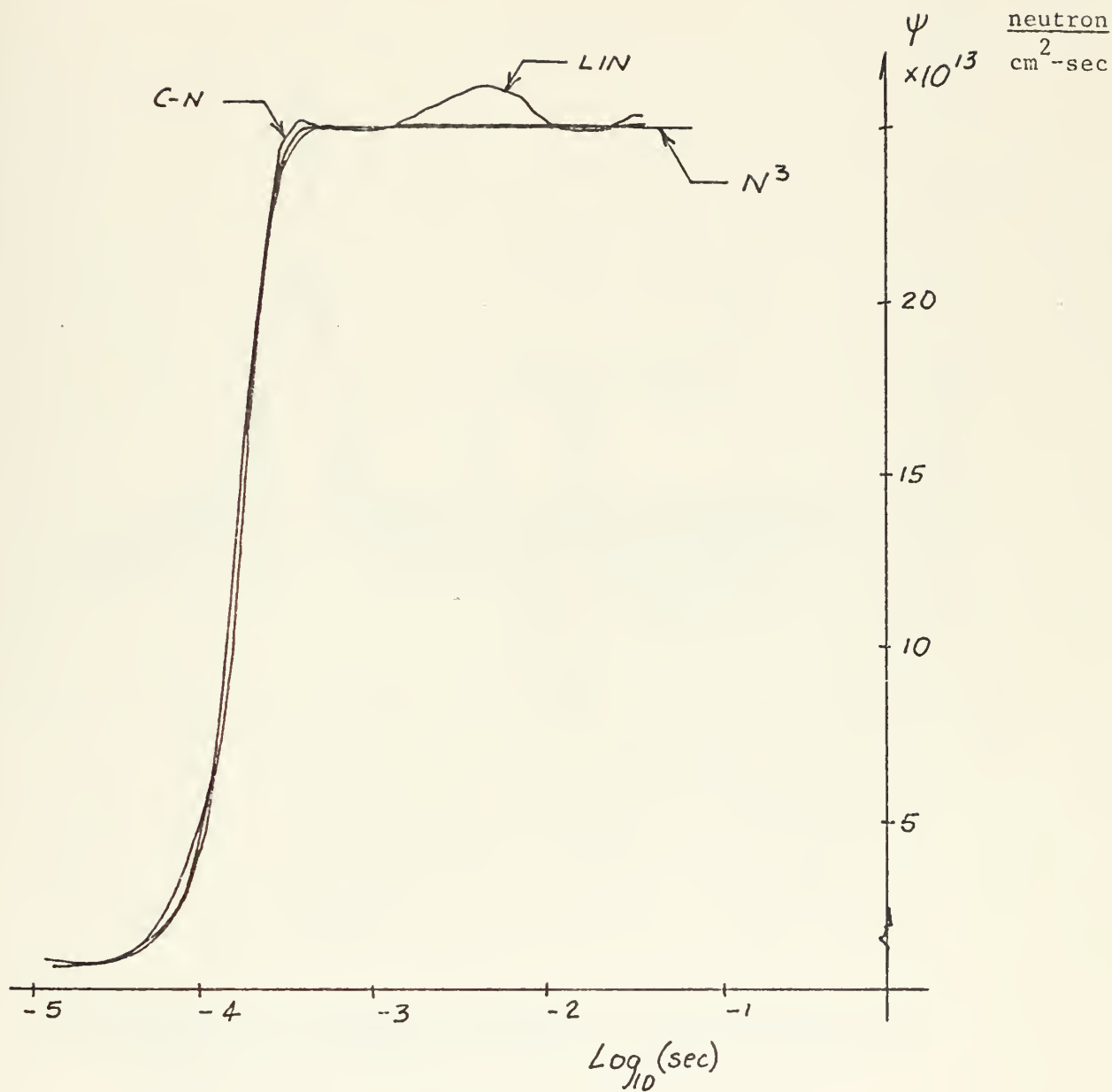


DEVIATION FROM  $N^3$  SOLUTION AT CORE POINT  $Z=40\text{cm}$ ,  $R=40\text{cm}$   
COMPARISON - UNIFORM DISTURBANCE - CORE POINT

FIGURE 13



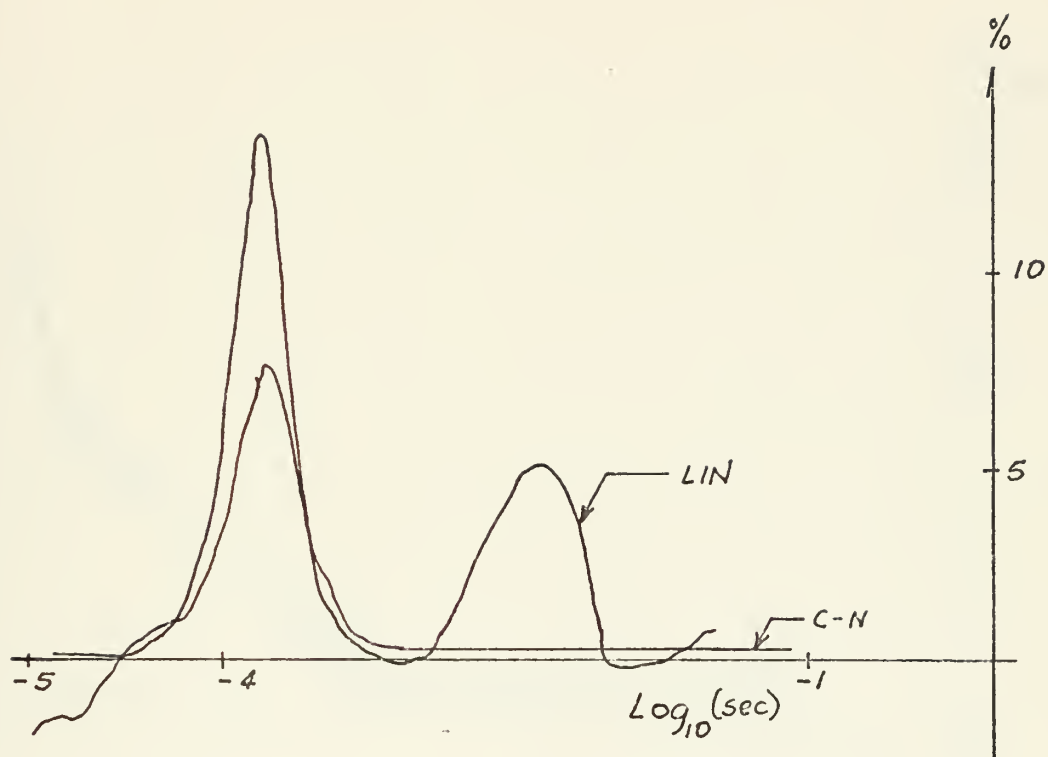




TIME DEPENDENT NEUTRON FLUX AT REFLECTOR POINT  $Z=80\text{cm}$ ,  $R=75\text{cm}$   
 FOR A UNIFORM DISTURBANCE  
 PLOT OF UNIFORM DISTURBANCE - REFLECTOR POINT

FIGURE 14

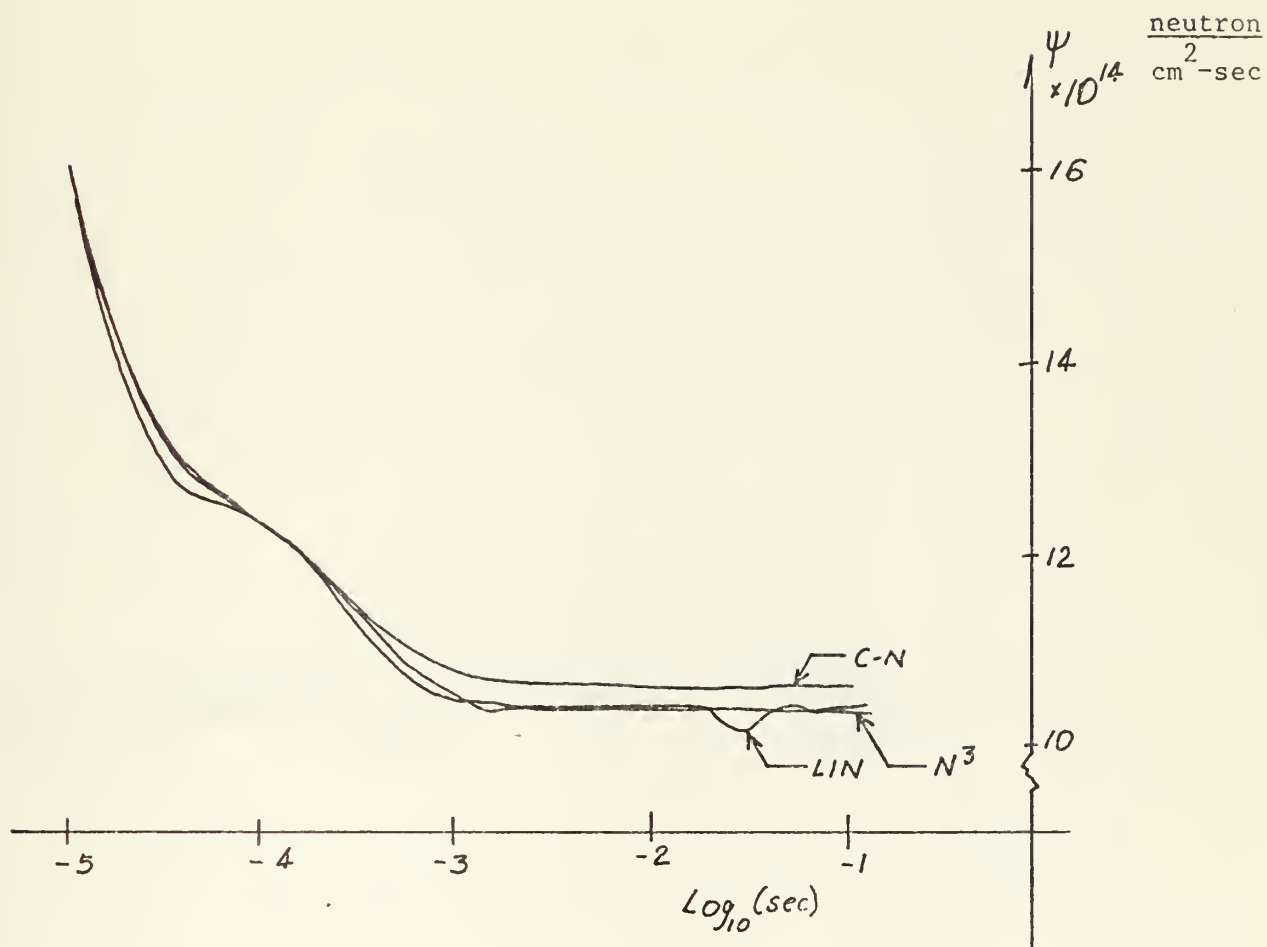




DEVIATION FROM  $N^3$  SOLUTION AT REFLECTOR POINT  $Z=80\text{cm}$ ,  $R=75\text{cm}$   
 COMPARISON - UNIFORM DISTURBANCE - REFLECTOR POINT

FIGURE 15

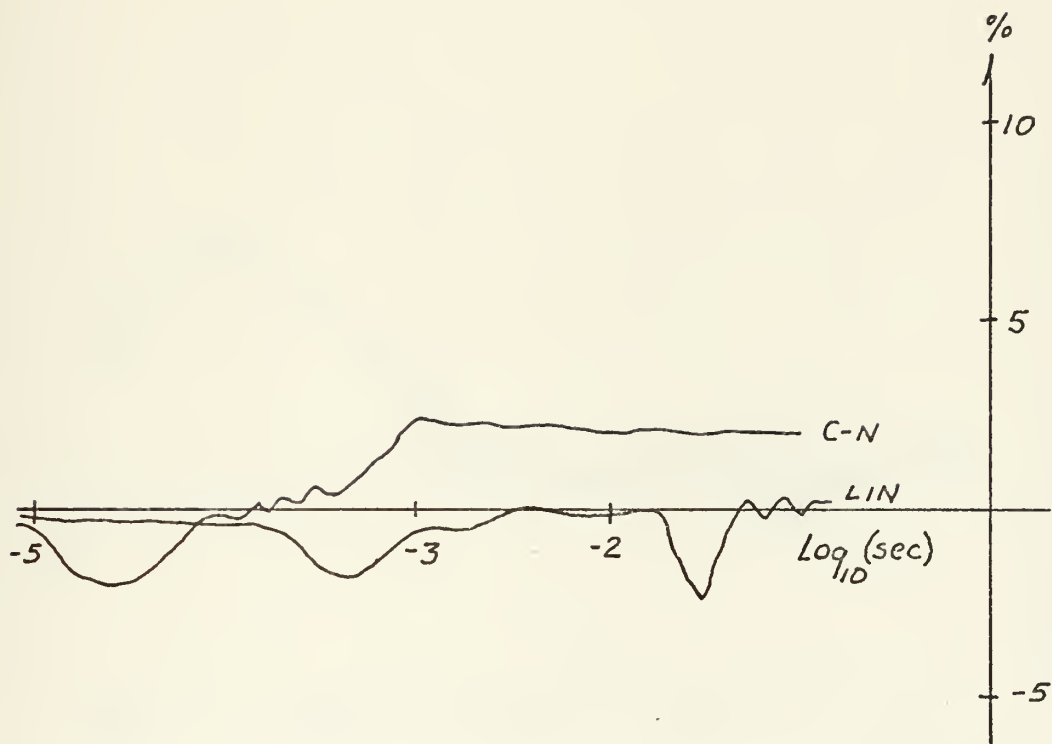




TIME DEPENDENT NEUTRON FLUX AT REACTOR CENTER  
FOR A SKEW DISTURBANCE  
PLOT OF SKEW DISTURBANCE - CENTER POINT

FIGURE 16



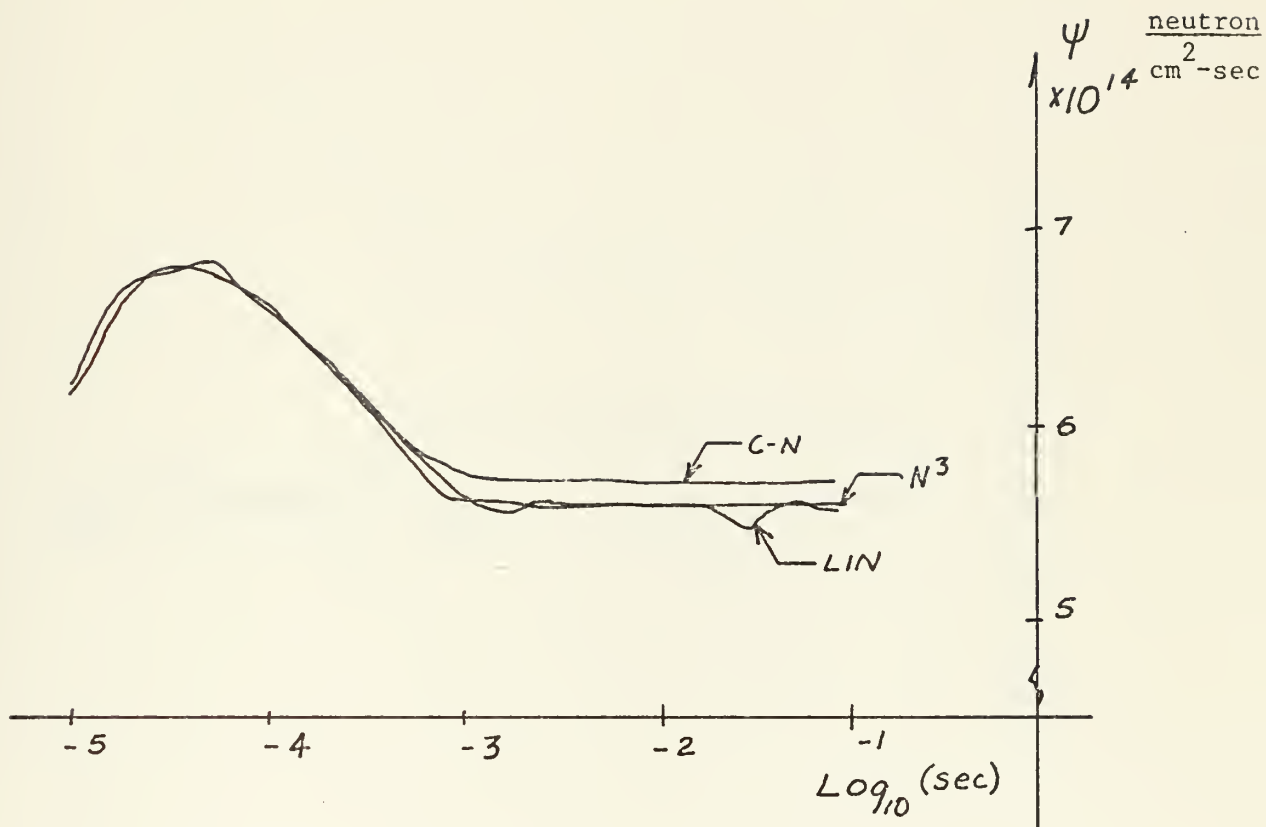


DEVIATION FROM  $N^3$  SOLUTION AT REACTOR CENTER  
COMPARISON - SKEW DISTURBANCE - CENTER POINT

FIGURE 17



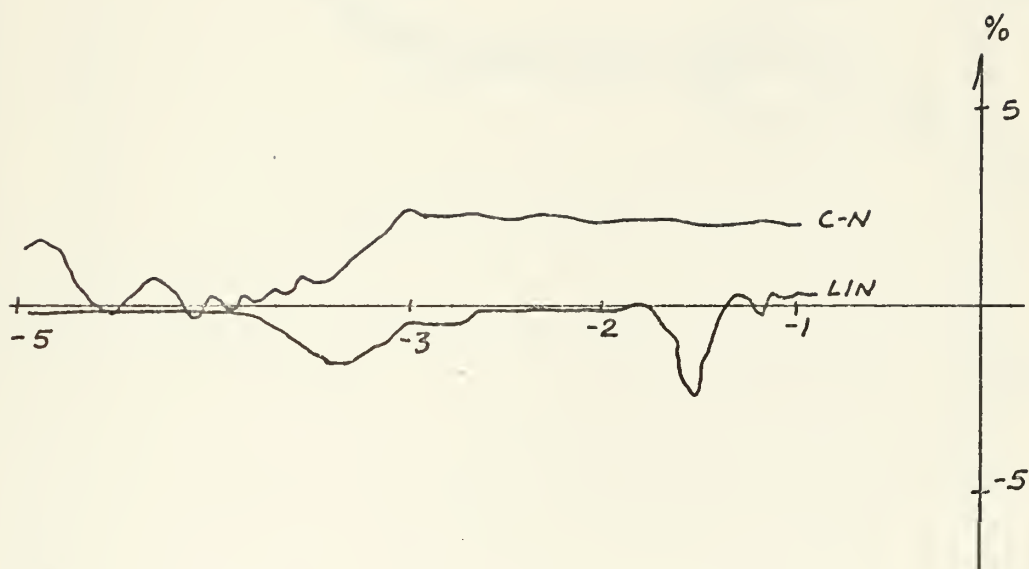




TIME DEPENDENT NEUTRON FLUX AT CORE POINT  $Z=40\text{cm}$ ,  $R=40\text{cm}$   
 FOR A UNIFORM DISTURBANCE  
 PLOT OF SKEW DISTURBANCE - CORE POINT

FIGURE 18

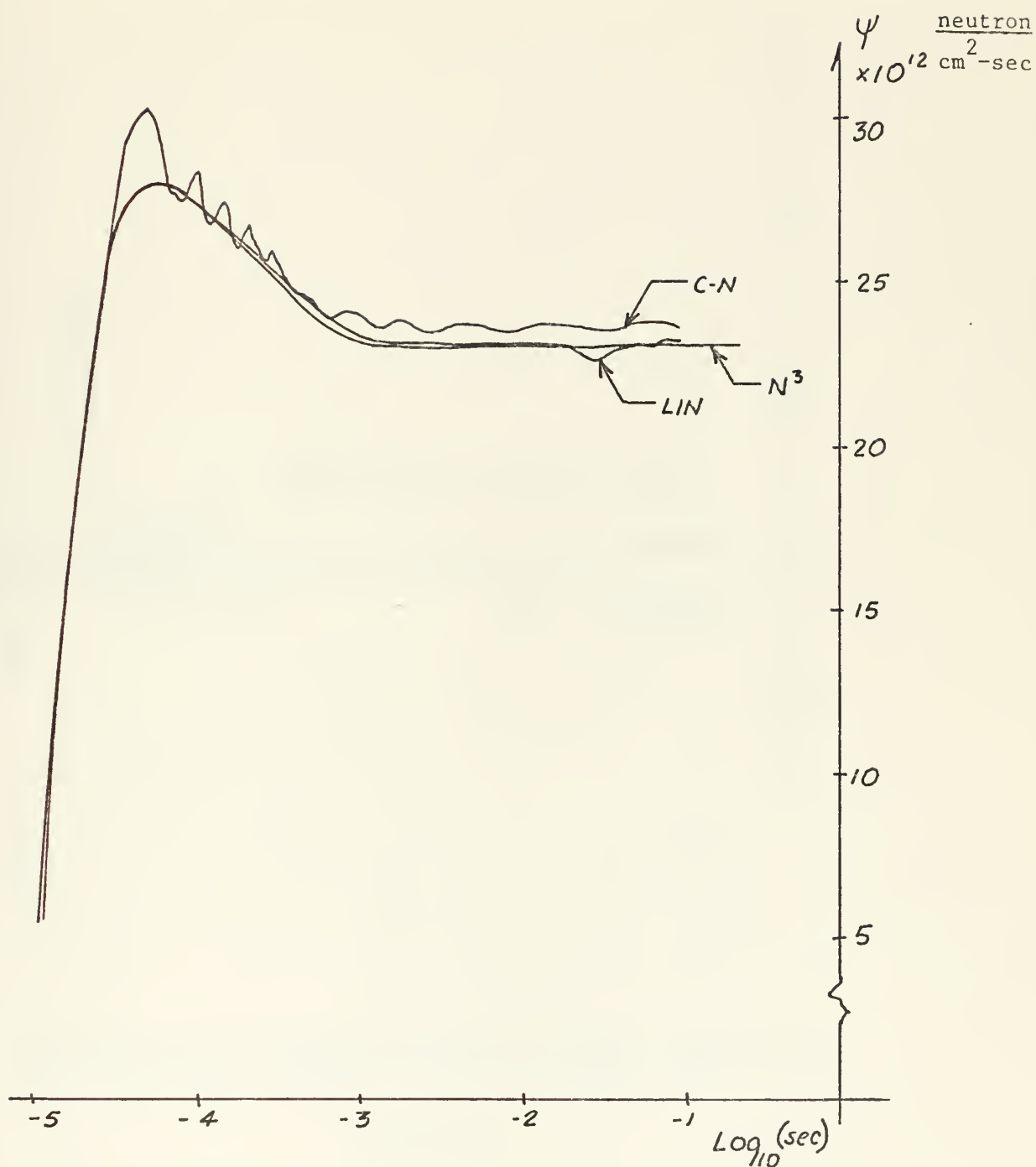




DEVIATION FROM  $N^3$  SOLUTION AT CORE POINT  $Z=40\text{cm}$ ,  $R=40\text{cm}$   
COMPARISON - SKEW DISTURBANCE - CORE POINT

FIGURE 19

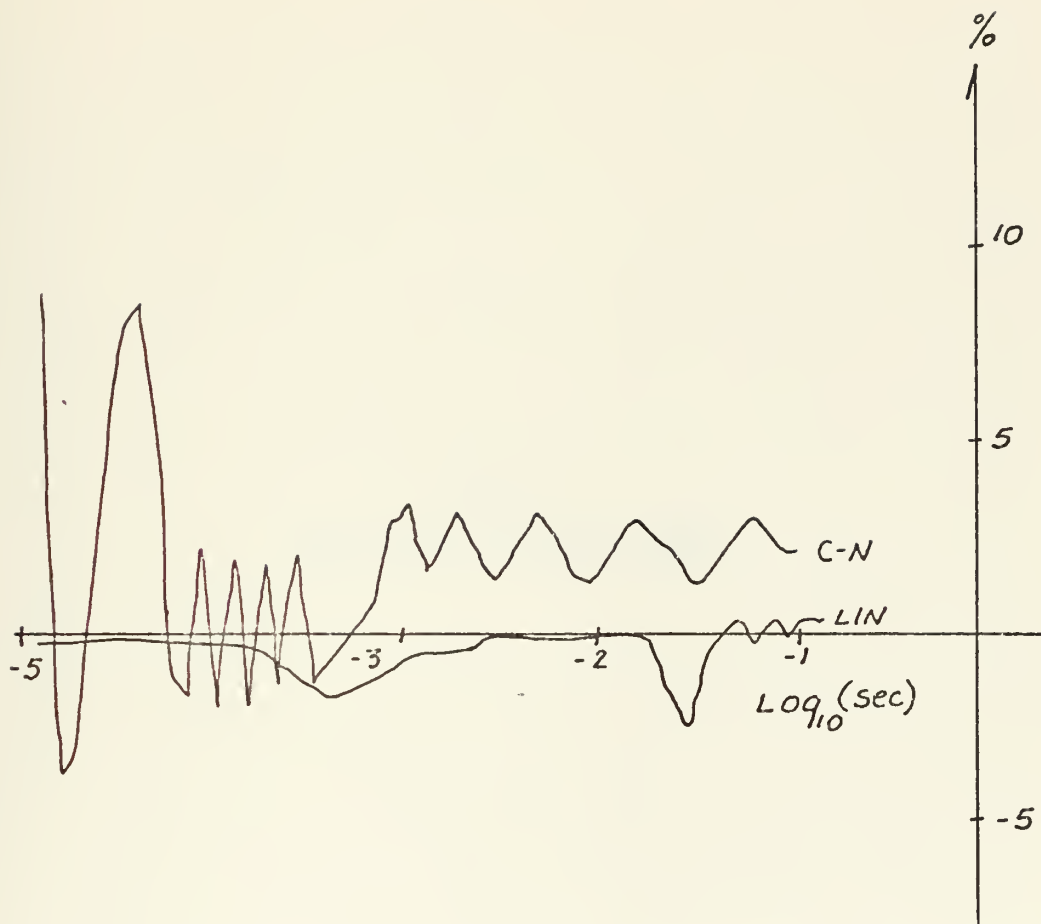




TIME DEPENDENT NEUTRON FLUX AT REFLECTOR POINT  $Z=80\text{cm}$ ,  $R=75\text{cm}$   
 FOR A SKEW DISTURBANCE  
 PLOT OF SKEW DISTURBANCE - REFLECTOR POINT

FIGURE 20





DEVIATION FROM  $N^3$  SOLUTION AT REFLECTOR POINT  $Z=80\text{cm}$ ,  $R=75\text{cm}$   
COMPARISON - SKEW DISTURBANCE - REFLECTOR POINT

FIGURE 21





- APPENDIX C  
COMPUTER PROGRAMMING CODES



## APPENDIX C

### COMPUTER PROGRAMMING CODES

- I. DISCUSSION OF PROCEDURES
- II. FED-2 LINEARIZED LISTING
- III. FED-2 COMPACT Nxp LISTING
- IV. DATA PROCESSOR LISTING
- V. COMPACTING DEMONSTRATION LISTING WITH RESULTS



## I. DISCUSSION

### A. The FED-2 Programs

Both of the FED-2 programs consist of a small calling program which initiates the problem. The rest of the program is stored in the computer library. The advantages of a pre-compiled main program in time and convenience is clear, but the use of dummy dimension statements to pass data storage throughout the calling of the external subroutine DVOGER and its user-supplied subroutine appears as a trick not normally considered possible with the FORTRAN language.

When the original  $N^3$  FED-2 program was written, every available device for reducing storage was attempted. Thus, where properties are normally summed over areas in the finite element method, the operations were carried out on the nodal points, since there are always fewer node points than elements. The new FED-2 programs continued these procedures not only to provide a valid comparison but also because this economizing technique is felt to be fundamentally sound even if unconventional. Employing this device, NFULEL is the number of node points associated with the core (fueled) region, whereas it would conventionally be the number of core (fueled) elements.

For these programs, three interest points IPT1, IPT2, and IPT3, may be selected for detailed investigation, since the purview of information produced by the printed output of these programs is quite large, making detailed study difficult. The time history of the neutron flux (PSI) at each of these interest points is recorded throughout the problem in an array BATCH. If it is desired to use this collection, it can be written onto a computer storage by the inclusion of additional JCL in the GO step. In this manner, data was saved for analysis by the data processing program used in this thesis to prepare the graphs of Figures 4 - 21.



An options chart and a schedule for FED-2 input data deck listing are included in this section in addition to a listing of the programs.





## OPTION SELECTIONS FOR FED-2 PROGRAMS

### NCOUNT

- = 0 PREMULTIPLICATION BY  $\psi_{t-\Delta t}$
- = 1 PREMULTIPLICATION BY  $\psi$  TRIAL

### MTH

- = 1 DVOGER, Jacobian must be supplied
- = 2 DVOGER, computes its own Jacobian
- = 0 DVOGER, doesn't use Jacobian
- = 5 CRANKO routine specified, DVOGER not activated

### ERRVAL

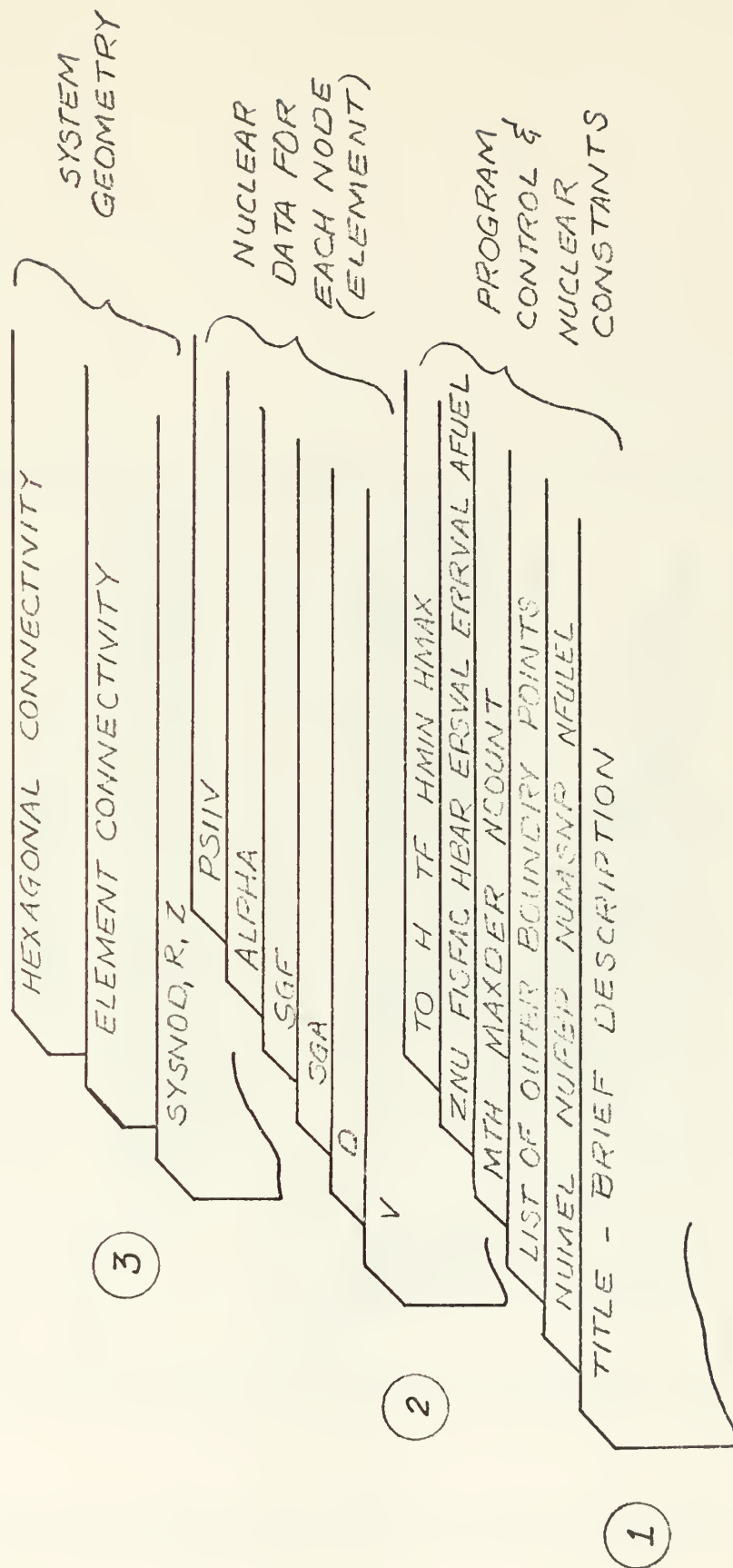
- 1- used internally by DVOGER, card input value has no effect
- 2- used by CRANKO, it is the convergence criterion for satisfactory  $\psi$  values

### EPSVAL

- 1- convergence criterion for DVOGER
- 2- for CRANKO, sets the relative problem time difference required for a solution to be printed out



# DATA DECK ARRANGEMENT





```

*****
**      PROGRAM FED-2-COMPACT
**
**      FINITE ELEMENT SOLUTION OF NON-LINEAR REACTOR
**      DYNAMICS IN TWO DIMENSIONAL SPACE WITH PRE-
**      MULTIPLICATION TO REDUCE THE NON-LINEARITY.
**
**      SOLUTION METHODS
**          1-DVOGER (IMSL)
**          2-CRANKO (INCLUDED)
**
**      OPTIONS
**          1- ALL OPTIONS AVAILABLE TO DVOGER ROUTINE
**             WITH SELECTION OF EITHER PREVIOUS TIME
**             OR CURRENT UPDATE
**          2- CRANKO ROUTINE BASED ON CRANK-NICOLSON
**             FORMULATION AND SOLVED BY GAUSS-SEIDEL
**             ITERATION.
**
**      INPUT PARAMETERS AND DIMENSIONING REQUIREMENTS.
**
**      SINGLE ITEMS
**      NUMSNP IS THE NUMBER OF SYSTEM NODAL POINTS.
**      NUMEL  IS THE NUMBER OF TRIANGULAR ELEMENTS.
**      NUPBP  IS THE NUMBER OF EXTERIOR BOUNDARY PTS.
**      EPSVAL IS THE CONVERGENCE CRITERIA FOR DVOGER.
**      ERRVAL IS THE CONVERGENCE CRITERIA FOR CRANKO
**      NCOUNT IS THE SELECTOR FOR UPDATE OR PREVIOUS
**              TIME FOR DVOGER.
**      MTH    IS THE OPTION SWITCH FOR CRANKO OR THE
**              SEVERAL DVOGER METHODS.
**      TO     IS THE PROBLEM START TIME
**      TF     IS THE PROBLEM FINISH TIME
**      H      IS THE FIRST ATTEMPTED TIME INTERVAL
**      HMIN   IS THE SMALLEST ALLOWED TIME INTERVAL
**      HMAX   IS THE LARGEST ALLOWED TIME INTERVAL
**      LCN    IS THE MAX NUMBER OF NODE CONTRIBUTORS
**
**      THESE ARE THE INPUT REQUIREMENTS
**
*****

```



```

*      *      *      *      *      *      *      *      *      *
*      TITLE      NUMEL NUPBP NUMSNP NFUEL IUPBP MTH
*      MAXORD      NCOUNT ZNU FISFAC HBAR EPSVAL ERRVAL
*      AFUEL      TO H TF HMIN HMAX V D SGA SGF ALPHA
*      PSIIV      SYSNOD R Z ELEMNT ELNOD ITYPE LCON MNOD
*
*      ITYPE=0      DENOTES CORE ELEMENTS
*      ITYPE=1      DENOTES BLANKET ELEMENTS
*
*      IPT1, IPT2, IPT3 ARE SELECTED INTEREST POINTS
*
*      ICON AND IPT1, IPT2, IPT3 ARE NOT ON DATA CARDS,
*      BUT ARE INSERTED DIRECTLY INTO THE CALLING PGM.
*
*

```

```

THESE ARE THE DIMENSIONING REQUIREMENTS.
PART(NUMSNP,LCON),BIGA(NUMSNP,LCON),BIGCC(NUMSNP,LCON),
BIGAB(NUMSNP,LCON),BIGC(NUMSNP,LCON),BIGCC(NUMSNP,LCON),
MNOD(NUMSNP,LCON),V(NUMSNP),D(NUMSNP),SGA(NUMSNP),
SGF(NUMSNP),ALPHA(NUMSNP),PSIIV(NUMSNP),PDPSI(NUMSNP)
ZLMBDA(NUMSNP),VLMBDA(NUMSNP),ZCMEGA(NUMSNP),
RHO(NUMSNP),SYSNOD(NUMSNP),R(NUMSNP),Z(NUMSNP),
ELEMNT(NUMEL),ELNOD(NUMEL,3),ITYPE(NUMEL),R1(NUMEL),
Z1(NUMEL),R2(NUMEL),Z2(NUMEL),R3(NUMEL),Z3(NUMEL)
A(NUMEL,3),B(NUMEL,3),PW(NWK),BATCH(400,4),
ERROR(NUMSNP,LCON),CM(NUMEL,3,3),ALFA(3,3),PSTEP(NUMSNP),
BIGD(NUMSNP,LCON),BICE(NUMSNP,LCON),ESTR(NUMSNP),VD(NUMSNP),
YMAX(NUMSNP),IUPBP(NUPBP),PSI(8,NUMSNP),DPSI(NUMSNP),
AREA(NUMEL),PV(NUMSNP),HOLD(NUMSNP)

```

#### DECLARATION STATEMENTS

```

NONLINEAR SUPERCRITICAL REACTOR
PRCMT FEEDBACK
FULLY REFLECTED SYSTEM
SPACE-DEPENDENT NEUTRONIC PROPERTIES

```

```

INTEGER*4 SYSNOD,ELEMNT,ELNOD

```





```

DIMENSION TITLE(20)

DIMENSION PART(132,7), BIGA(132,7), BIGAB(132,7), BIGC(132,7),
BIGCC(132,7), MNOD(132,7), V(132,7), D(132,7), SGA(132,7), SGF(132,7),
ALPHA(132,7), PSIIV(132,7), PDPSI(132,7), ZLMBDA(132,7), VLMBDA(132,7),
ZOMEGA(132,7), RHO(132,7), SYSNGD(132,7), R(132,7), Z(132,7), ELEMNT(220,7),
ELNOD(220,3), ITYPE(220,3), RI(220,3), R2(220,3), Z2(220,3),
R3(220,3), Z3(220,3), A(220,3), PW(2244,3), BATICH(400,4), ERROR(132,7),
CM(220,3,3,3), PSTEP(132,7), BIGD(132,7), BIGE(132,7), ESTR(132,7),
VD(132,7), YMAX(132,7), IUPBP(22,7), PSI(8,132), AREA(220,7), PV(132,7),
B(220,3), ASTR(132,7), HOLD(132)

READ(5,998) TITLE
READ(5,10) NUMEL,NUPBP,NUMSNP,NFULEL
FCRMMAT(20A4)
10 FORMAT(8I10)

FOR MTH=1 NWK MUST BE DIMENSIONED AS NUMSNP*(NUMSNP+17)
LCCN=7

IPT1=1
IPT2=41
IPT3=91

* CALL FEDMAN (NUMEL,NUPBP,NUMSNP,NFULEL,LCON,NWK,TITLE,
1 IPT1,IPT2,IPT3,PART,BIGA,BIGAB,BIGC,BIGCC,MNOD,V,D,SGA,SGF,
2 ALPHA,PSIIV,PDPSI,ZLMBDA,VLMBDA,ZOMEGA,RHO,SYSNOD,R,Z,ELEMNT,
3 ELNOD,ITYPE,R1,Z1,R2,Z2,R3,Z3,A,B,PW,BATCH,ERROR,CM,ALFA,
4 PSTEP,BIGD,BIGE,ASTR,ESTR,VD,YMAX,DPSI,IUPBP,PSI,AREA,PV,
5 HOLD)
* STOP
END
SUBROUTINE FEDMAN (NUMEL,NUPBP,NUMSNP,NFULEL,LCON,NWK,TITLE,
1 IPT1,IPT2,IPT3,PART,BIGA,BIGAB,BIGC,BIGCC,MNOD,V,D,SGA,SGF,
2 ALPHA,PSIIV,PDPSI,ZLMBDA,VLMBDA,ZOMEGA,RHO,SYSNOD,R,Z,ELEMNT,
3 ELNOD,ITYPE,R1,Z1,R2,Z2,R3,Z3,A,B,PW,BATCH,ERROR,CM,ALFA,
4 PSTEP,BIGD,BIGE,ASTR,ESTR,VD,YMAX,DPSI,IUPBP,PSI,AREA,PV,
5 HOLD)
* STOP
END
SUBROUTINE FINITE ELEMENT SOLUTION OF NONLINEAR
***

```



```

C      **      REACTOR DYNAMICS IN TWO-DIMENTIONAL SPACE      **
C      **      **
C      **      INTEGER*4  SYSNGD,ELEMNT,ELNOD      **
C      **      DIMENSION TITLE(20)      **
C
C      DIMENSION PART(NUMSNP,LCON),BIGA(NUMSNP,LCON),
C      1      BIGAB(NUMSNP,LCON),BIGC(NUMSNP,LCON),RIGCC(NUMSNP,LCON),
C      2      MNOD(NUMSNP,LCON),V(NUMSNP),D(NUMSNP),SGA(NUMSNP),
C      3      SGF(NUMSNP),ALPHA(NUMSNP),PSIIV(NUMSNP),PDPSI(NUMSNP)
C      DIMENSION ZLMBDA(NUMSNP),VLMBDA(NUMSNP),ZOMEGA(NUMSNP),
C      1      RHO(NUMSNP),SYSNOD(NUMSNP),R(NUMSNP),Z(NUMSNP),
C      2      ELEMNT(NUMEL),ELNOD(NUMEL,3),ITYPE(NUMEL),R1(NUMEL),
C      3      Z1(NUMEL),R2(NUMEL),Z2(NUMEL),R3(NUMEL),Z3(NUMEL)
C      DIMENSION A(NUMEL,3),B(NUMEL,3),PW(NWK),BAT(400,4),
C      1      ERROR(NUMSNP),CM(NUMEL,3,3,3),ALFA(3,3),PSTEP(NUMSNP),
C      2      BIGD(NUMSNP,LCON),BIGE(NUMSNP,LCON),ESTR(NUMSNP),VD(NUMSNP),
C      3      YMAX(NUMSNP),IUPBP(NUPBP),PSI(8,NUMSNP),DPSI(NUMSNP),
C      4      AREA(NUMEL),PV(NUMSNP),HOLD(NUMSNP)
C
C      DO 05 I=1,NUMSNP
C      ERROR(I)=0.
C      POPSI(I)=0.
C      YMAX(I)=1.
C      05 CCNTINUE
C
C      NLF=NUMSNP/5
C      XNP=NUMSNP
C      XNUF=XNP/5.
C      IF ((XNUF-FLOAT(NUF)) .LT. 1.) NUF=NUF+1
C
C      GENERAL PROBLEM DATA
C
C      READ(5,12) (IUPBP(I),I=1,NUPBP)
C      READ(5,10) MTH,MAXDER,NCOUNT
C      10  FORMAT(8I10)
C      12  FORMAT(16I5)
C
C      WRITE(6,11) NUMEL,NUPBP,NUMSNP,NFUEL
C      11  FORMAT(/2X,'# OF ELEMENTS=',I5//2X,'# OF UPPER BDRY NODAL POINTS=',
C      215)
C      15//2X,'# OF SYSTEM NODAL POINTS=',I5//2X,'# OF FUEL ELEMENTS=',
C
C      READ(5,15) ZNU,FISFAC,HBAR,EPSVAL,ERRVAL,AFUEL
C      READ(5,19) TO,H,TF,HMIN,HMAX

```













```

19X,'Z3'//)
C
DO 100 I=1,NUMEL
C
J=ELNOD(I,1)
R1(I)=R(J)
Z1(I)=Z(J)
C
K=ELNOD(I,2)
R2(I)=R(K)
Z2(I)=Z(K)
C
L=ELNOD(I,3)
R3(I)=R(L)
Z3(I)=Z(L)
C
WRITE (6,105) I,R1(I),Z1(I),R2(I),Z2(I),R3(I),Z3(I)
C
100 CONTINUE
105 FORMAT (3X,I3,3X, 7(F10.6,1X))
C
C COMPUTE A1,A2,A3,B1,B2,B3,AREA FOR EACH ELEMENT
C-----
C
WRITE (6,120)
120 FORMAT (1X//1X,'ELEMENT',5X,'A1',9X,'A2',9X,'A3',9X,'B1',9X,'B2',
19X,'B3',8X,'AREA'//)
C
DO 140 I=1,NUMEL
A(I,1)=R3(I)-R2(I)
A(I,2)=R1(I)-R3(I)
A(I,3)=R2(I)-R1(I)
B(I,1)=Z2(I)-Z3(I)
B(I,2)=Z3(I)-Z1(I)
B(I,3)=Z1(I)-Z2(I)
AREA(I)=0.5*(A(I,2)*B(I,1)-A(I,1)*B(I,2))
C
WRITE(6,130) I,A(I,1),A(I,2),A(I,3),B(I,1),B(I,2),B(I,3),
1AREA(I)
140 CONTINUE
C
130 FORMAT (3X,I3,3X, 7(F12.7,1X))
C
C----- ZERO THE BIGA, BIGC AND BIGAB MATRICES-----
C
C
C
C
C

```



```

C      DO 194 KK=1,NUMSNP
C      DO 193 II=1,LCON
C      BIGA(KK,II)=0.
C      BIGAB(KK,II)=0.0
C      BIGC(KK,II)=0.0
C      PART(KK,II)=0.0
C      193 CONTINUE
C      194 CONTINUE
C      *****
C      -----
C      PEPSI CALCULATES THE BIGA MATRIX AND PART OF THE BIGAB
C      -----
C      CALL PEPSI (NUMEL,NUMSNP,R1,R2,R3,AREA,ELNOD,LCON,MNOD,
C      1 BIGA,BIGAB,VLMBDA)
C      195 FORMAT(/5X,'BIGA MATRIX',//)
C      WRITE(6,195)
C      778 WRITE(6,778) ((BIGA(I,J),J=1,LCON),I=1,NUMSNP)
C      778 FORMAT(2X,7(E14.5))
C      196 FORMAT(/5X,'BIGAB MATRIX FROM PEPSI, NO DEL**2 CONTRIBUTION',//)
C      WRITE(6,778) ((BIGAB(I,J),J=1,LCON),I=1,NUMSNP)
C      -----
C      CRUISE CALCULATES THE BIGAB MATRIX
C      -----
C      CALL CRUISE (NUMEL,NUMSNP,LCON,R1,R2,R3,AREA,ELNOD,
C      1 BIGAB,MNOD,A,B,VD)
C      197 WRITE(6,197)
C      197 FORMAT(/5X,'BIGAB MATRIX FROM CRUISE, WITH DEL**2 TERM',//)
C      WRITE(6,778) ((BIGAB(I,J),J=1,LCON),I=1,NUMSNP)
C      -----
C      DO 255 I=1,NUPBP
C      II=IUPBP(I)
C      DO 251 J=1,NUMSNP
C      BIGAB(II,I)=0.0

```







```

C      PTIME= H
      WRITE(6,999) TITLE
999  FORMAT(1H1,20A4)
      IF (MTH .EQ. 0) WRITE (6,33)
      IF (MTH .EQ. 1) WRITE (6,31)
      IF (MTH .EQ. 5) WRITE (6,35)
31  FORMAT (10X, 'THIS PGM UTILIZES GEERS METHOD IN DOVGER')
33  FORMAT (10X, 'THIS PGM UTILIZES ADAMS PERDICTOR-CORRECTOR ',
1  'METHOD IN DOVGER')
35  FORMAT (10X, 'THIS PGM UTILIZES CRANK-NICOLSON METHOD TO SET',
1  'UP A SET OF SIMULTANEOUS LINEAR EQNS WHICH ARE THEN',
2  ',/10X, 'SOLVED BY SUCCESSIVE ITERATION WITH IMPROVEMENT.')
C
      WRITE(6,317)
317  FORMAT (//10X, 'INITIAL ARGUMENTS',//)
      IF (MTH .EQ. 5) GO TO 319
      WRITE(6,318) I, MTH, MAXDER, JSTART, H, HMIN, HMAX, EPS, NCOUNT
318  FORMAT(2X, 'I=', G10.4, '2X, 'MTH=', I4, '2X, 'MAXDER=', I4, '2X, 'JSTART=', I4,
1  '2X, 'H=', G10.4, '2X, 'HMIN=', G10.4, '2X, 'HMAX=', G10.4, '2X, 'EPS=', G10.4, '2X,
2  'NCOUNT=', I2, '/')
319  CCNTINUE
C
      DO 320 I=1, NUMSNP
      ERROR(I)= ERRVAL
      YMAX(I)=1.
      PSI(I, I)=PS11V(I)
320  CONTINUE
C
      DO 321 I=1, NUPBP
      J=IUPBP(I)
      PSI(I, J)=0.0
321  CONTINUE
C
      NUMEQ=NUMSNP-NUPBP
C
C
C
C
C-----
C      CCNSTRUCT THE BIGC MATRIX ON THE ELEMENT LEVEL
C
C      DO 200 L=1, NUMEL
C
C      DO 583 I=1, 3
C      DO 582 J=1, 3
C      DO 581 K=1, 3
C      CM(L, I, J, K) =0.

```





```

581 CONTINUE
582 CONTINUE
583 CONTINUE
LL=ITYPE(L)
IF (LL.NE. 0) GO TO 200
C
C=PI*AREA(L)/180.
CM(L,3,3,3)=CC*(6.0*R1(L)+6.0*R2(L)+24.*R3(L))
CM(L,3,3,2)=CC*(2.0*R1(L)+4.0*R2(L)+6.0*R3(L))
CM(L,3,3,1)=CC*(4.0*R1(L)+2.0*R2(L)+6.0*R3(L))
CM(L,3,2,2)=CC*(2.0*R1(L)+4.0*R2(L)+4.0*R3(L))
CM(L,3,2,1)=CC*(2.0*R1(L)+2.0*R2(L)+2.0*R3(L))
CM(L,3,1,3)=CC*(4.0*R1(L)+2.0*R2(L)+6.0*R3(L))
CM(L,3,1,2)=CC*(2.0*R1(L)+4.0*R2(L)+4.0*R3(L))
CM(L,3,1,1)=CC*(6.0*R1(L)+2.0*R2(L)+4.0*R3(L))
CM(L,2,3,3)=CC*(2.0*R1(L)+6.0*R2(L)+4.0*R3(L))
CM(L,2,3,2)=CC*(2.0*R1(L)+2.0*R2(L)+2.0*R3(L))
CM(L,2,3,1)=CC*(2.0*R1(L)+4.0*R2(L)+6.0*R3(L))
CM(L,2,2,2)=CC*(6.0*R1(L)+2.0*R2(L)+2.0*R3(L))
CM(L,2,2,1)=CC*(4.0*R1(L)+6.0*R2(L)+4.0*R3(L))
CM(L,2,1,3)=CC*(2.0*R1(L)+2.0*R2(L)+2.0*R3(L))
CM(L,2,1,2)=CC*(4.0*R1(L)+6.0*R2(L)+2.0*R3(L))
CM(L,2,1,1)=CC*(6.0*R1(L)+2.0*R2(L)+6.0*R3(L))
CM(L,1,3,3)=CC*(4.0*R1(L)+2.0*R2(L)+2.0*R3(L))
CM(L,1,3,2)=CC*(2.0*R1(L)+4.0*R2(L)+6.0*R3(L))
CM(L,1,3,1)=CC*(6.0*R1(L)+2.0*R2(L)+4.0*R3(L))
CM(L,1,2,2)=CC*(2.0*R1(L)+6.0*R2(L)+2.0*R3(L))
CM(L,1,2,1)=CC*(4.0*R1(L)+2.0*R2(L)+4.0*R3(L))
CM(L,1,1,3)=CC*(6.0*R1(L)+4.0*R2(L)+2.0*R3(L))
CM(L,1,1,2)=CC*(2.0*R1(L)+2.0*R2(L)+6.0*R3(L))
CM(L,1,1,1)=CC*(24.*R1(L)+6.0*R2(L)+6.0*R3(L))
C 200 CONTINUE
C 151 CONTINUE
C
C
IF (MTH.EQ.5) CALL CRANKO (NUMEL,NUMSNP,NUPBP,LCON,H,TO,
1 ERRVAL,EPVAL,IUPBP,PSIIV,ITYPE,ELNOD,CM,MNOD,ZOMEGA,
2 BIGC,BIGA,IPT1,IPT2,IPT3,PV,PSTEP,BIGD,BIGE,ASTR,ESTR,
3 BIGC,BIGCC,HOLD,TF)
C
C
EXTERNAL YVETTE

```



```

C      CALL YVET (NUMEL, NUMEQ, NUMSNP, LCON, IUPBP, PSIIV, NUPBP, BIGC,
1      IYPE, ELNOD, CM, BIGA, BIGAB, MNOD, BIGCC, PART, NIK, PSI, PW, DPSI,
2      PPSI, HOLD, DPT, PV, NCOUNT, ZOMEGA)
C
C      351 CONTINUE
C
C      IF (NCOUNT .EQ. 1) GO TO 312
C
C      *****
C      FIXES THE NEW BIGCC BY PREMULTIPLYING BY PSI(T-1)
C      *****
C
C      DO 210 L=1, NUMEL
C      LL=ITYPE(L)
C      IF (LL .NE. 0) GO TO 210
C      N1=ELNOD(L,1)
C      N2=ELNOD(L,2)
C      N3=ELNOD(L,3)
C      DO 88 J=1,3
C      DO 90 N=1,3
C      ALFA(N,J)=PSIIV(N1)*CM(L,N,J,1)+PSIIV(N2)*CM(L,N,J,2)
C      +PSIIV(N3)*CM(L,N,J,3)
C      1 CONTINUE
C      90 CONTINUE
C      88 CONTINUE
C      DO 150 K=1,3
C      KK=ELNOD(L,K)
C      DO 180 I=1,3
C      II=ELNOD(L,I)
C      DO 91 MX=1, LCON
C      NOW =MX
C      MM = MNOD(KK, MX)
C      IF (MM .EQ. II) GO TO 92
C      91 CONTINUE
C      92 CONTINUE
C      BIGC(KK, NOW)=BIGC(KK, NOW)+ALFA(K,I)*ZOMEGA(KK)
C      180 CONTINUE
C      150 CONTINUE
C      210 CONTINUE
C      WRITE(6,755)
C      755 FORMAT(///10X, ' BIGC MATRIX BEFORE BOUNDARY POINTS' /)
C      WRITE(6,778)((BIGC(I,J), J=1, LCON), I=1, NUMSNP)
C
C      INSERTION OF BOUNDARY POINTS
C      DO 285 I=1, NUPBP
C      II=IUPBP(I)
C      DO 291 MX=1, LCON
C      BIGC(II, MX)=0

```



```

291 CONTINUE
285 CCNTINUE
C 757 WRITE(6,757)
C 757 FORMAT(//10X,'BIGC MATRIX WITH BOUNDARY POINTS',/)
C 757 WRITE(6,773) ((BIGC(I,J),J=1,7),I=1,NUMSNP)
C
C
C 310 I=1,NUMSNP
C 310 J=1,LCON
C 310 BIGCC(I,J)=BIGAB(I,J)-BIGC(I,J)
C 310 PART(I,J)=BIGAB(I,J)-2.*BIGC(I,J)
C 310 CONTINUE
C 759 WRITE(6,759)
C 759 FORMAT(//10X,'BIGCC MATRIX',/)
C 759 WRITE(6,778) ((BIGCC(I,J),J=1,LCON),I=1,NUMSNP)
C
C
312 CONTINUE
C
C
C *****
C * SOLVE NON-LINEAR FLUX EQUATIONS WITH SUBROUTINE DVOGER *
C *****
C
350 CALL DVOGER(YVETTE,PSI,T,NUMSNP,MTH,MAXDER,JSTART,H,HMIN,HMAX,EPS,
1YMAX,ERROR,PW,IER)
C
C
C IF(IER.EQ.0) GO TO 3540
C JSTART=-1
C H=HMIN*.1
C HMIN=H*.1
C GO TO 350
C
C 3540 CCNTINUE
C JSTART=1
C IF((T-PTIME)/PTIME .LT. STAR) GO TO 362
C PTIME=T
C 3510 WRITE(6,3510)T,H,JSTART,IER
C 3510 FORMAT(//2X,'T=',G12.6,2X,'H=',G12.6,2X,'JSTART=',I4,2X,'IER=',I4)
C 354 WRITE(6,354)
C 354 FORMAT(//5(4X,'NODE',7X,'PSI',4X))
C
C
C NIT= 1 + NIT
C IF (NIT.GT. 400) GO TO 377
C BATCH (NIT,1)=T

```



```

BATCH(NIT,2)=PV(IPT1)
BATCH(NIT,3)=PV(IPT2)
BATCH(NIT,4)=PV(IPT3)
377 CONTINUE
C
DO 355 I=1,NUF
  I1=I+NUF
  I2=I+2*NUF
  I3=I+3*NUF
  I4=I+4*NUF
  WRITE(6,356) (I,PSI(1,I),I1,PSI(1,I1),I2,PSI(1,I2),I3,PSI(1,I3),I4,
1,PSI(1,I4))
355 CONTINUE
356 FORMAT(5(4X,I3,3X,1PE12.4))
C
DO 360 I=1,NUMSNP
  R(I)=PSI(2,I)/H
360 CONTINUE
C
WRITE(6,357)
FORMAT(/(5(4X,'NODE',7X,'DPSI',3X)))
DO 359 I=1,NUF
  I1=I+NUF
  I2=I+2*NUF
  I3=I+3*NUF
  I4=I+4*NUF
  WRITE(6,356) (I,R(I),I1,R(I1),I2,R(I2),I3,R(I3),I4,R(I4))
359 CCNTINUE
362 CONTINUE
C
DO 27 MA=1,NUMSNP
  PSIIV(MN)=PSI(1,MN)
DO 28 I=1,LCON
  BIGC(MN,I)=.0
28 CONTINUE
27 CONTINUE
C
IF (I.LT.TF) GO TO 351
C
C
399 FORMAT(10X,I4)
WRITE(6,399) NIT
WRITE(3) BATCH
WRITE(4) BATCH
STOP
END
SUBROUTINE PEPSI (NUMEL,NUMSNP,R1,R2,R3,AREA,ELNOD,LCON,MNOD,
1 BIGA,BIGAR,VLMBDA)

```









```

C 200 CONTINUE
C
C RETURN
C END
C SUBROUTINE CRUISE (NUMEL,NUMSNP,LCON,R1,R2,R3,AREA,ELNOD,BIGAB,
C 1 MNOD,A,B,VD)
C *
C-----
C CRUISE CALCULATES THE BIGAB MATRIX
C-----
C
C INTEGER*4 ELNOD
C
C DIMENSION R1(NUMEL),R2(NUMEL),R3(NUMEL),AREA(NUMEL),
C 1 MNOD(NUMSNP,LCON),BIGAB(NUMSNP,LCON),BMATRIX(3,3),
C 2 ELNOD(NUMEL,3),A(NUMEL,3),B(NUMEL,3),VD(NUMSNP)
C
C PI=3.1415927
C
C-----
C CALCULATE THE 3X3 B(I,J) MATRIX FOR ELEMENT L
C-----
C
C DO 200 L=1,NUMEL
C COEFFB= PI / (6.0 * AREA(L) )
C
C BMATRIX(1,1)=COEFFB *(-R1(L))*(2.0*B(L,1)**2+2.0*A(L,1)**2)-R2(L)*
C 1(B(L,1)*B(L,2)+B(L,1)**2+A(L,1)*A(L,2)+A(L,1)**2)*B(L,1)*
C 2B(L,3)+B(L,1)**2+A(L,1)*A(L,3)+A(L,1)**2)+2.0*AREA(L)*B(L,1)}
C
C BMATRIX(1,2)=COEFFB *(-R1(L))*(2.0*B(L,1)*B(L,2)+2.0*A(L,1)*
C 1A(L,2))-R2(L)*(B(L,2)*B(L,1)+B(L,2)+A(L,1)*A(L,2))-
C 2R3(L)*(B(L,2)*B(L,3)+B(L,1)*B(L,2)+A(L,2)+A(L,1)*A(L,2))+
C 32.0*AREA(L)*B(L,2)}
C
C BMATRIX(1,3)=COEFFB *(-R1(L))*(2.0*B(L,1)*B(L,3)+2.0*A(L,1)*
C 1A(L,3))-R2(L)*(B(L,2)*B(L,1)+B(L,3)+A(L,2)*A(L,1)+A(L,1)*
C 2A(L,3))-R3(L)*(B(L,3)*B(L,1)+B(L,3)+A(L,3)+A(L,1)*A(L,3))+
C 32.0*AREA(L)*B(L,3)}
C
C BMATRIX(2,1)=BMATRIX(1,2)
C
C BMATRIX(2,2)=COEFFB *(-R1(L))*(B(L,2)*B(L,1)*B(L,2)+A(L,2)**2+
C 1A(L,1)*A(L,2))-R2(L)*(2.0*B(L,2)**2+2.0*A(L,2)**2)-R3(L)*(B(L,3)*
C 2B(L,2)+B(L,2)**2+A(L,3)*A(L,2)+A(L,2)**2)+2.0*AREA(L)*B(L,2)}

```



```

BMATRIX(2,3)=COEFFB *(-R1(L)*(B(L,2)*B(L,3)+B(L,1)*B(L,3)+A(L,2))*
1A(L,3)+A(L,1)*A(L,3))-R2(L)*(2.0*B(L,2)*B(L,3)+2.0*A(L,2)*A(L,3))-
2R3(L)*(B(L,3)**2+B(L,2)*B(L,3)+A(L,3)**2+A(L,2)*A(L,3))+
32.0*AREA(L)*B(L,3))

```

```

C
BMATRIX(3,1)=BMATRIX(1,3)

```

```

C
BMATRIX(3,2)=BMATRIX(2,3)

```

```

C
BMATRIX(3,3)=COEFFB *(-R1(L)*(B(L,3)**2+B(L,1)*B(L,3)+A(L,3)**2+
1A(L,1)*A(L,3))-R2(L)*(B(L,3)**2+B(L,2)*B(L,3)+A(L,3)**2+A(L,2)*
2A(L,3))-R3(L)*(2.0*B(L,3)**2+2.0*A(L,3)**2)+2.0*AREA(L)*B(L,3))

```

```

C
C STORE ELEMENT MATRIX, AMATRIX, INTO SYSTEM MATRIX, BIGA
C-----

```

```

DC 20 K=1,3

```

```

KK=ELNOD(L,K)

```

```

DC 10 I=1,3

```

```

II=ELNOD(L,I)

```

```

DO 91 MX=1,LCON

```

```

NCW=MX

```

```

MM=MNOD(KK,MX)

```

```

IF (MM.EQ. II) GO TO 92

```

```

91 CONTINUE

```

```

92 CONTINUE

```

```

BIGAB(KK,NOW)=BIGAB(KK,NOW)+VD(KK)*BMATRIX(K,I)

```

```

10 CONTINUE

```

```

20 CONTINUE

```

```

C 200 CONTINUE

```

```

C
RETURN

```

```

C
END

```

```

SUBROUTINE YVET(NUMEL,NUMEQ,NUMSNP,LCON,IUPBP,PSIIV,NUPBP,BIGC,

```

```

1 ITYPE,ELNOD,CM,BIGA,BIGAB,MNOD,BIGCC,PART,NWK,PSI,PW,DPSI,

```

```

2 PDPSI,HOLD,DPT,PV,NCOUNT,ZOMEGA)

```

```

C
C
INTEGER*4 ELNOD

```

```

C
DIMENSION IUPBP(NUPBP),PSIIV(NUMSNP),BIGC(NUMSNP,LCON),

```

```

1 ELNOD(NUMEL,3),CM(NUMEL,3,3),BIGA(NUMSNP,LCON),

```

```

2 BIGAB(NUMSNP,LCON),MNOD(NUMSNP,LCON),BIGCC(NUMSNP,LCON),

```

```

3 PART(NUMSNP,LCON),PDPSI(NUMSNP)

```

```

C

```



```

DIMENSION HOLD(NUMSNP),PSI(8,NUMSNP),DPT(NUMSNP),DPSI(NUMSNP),
1 PW(NWK),PV(NUMSNP),ALFA(3,3)
RETURN
C
C *****
C
C ENTRY YVETTE(PSI,T,M,DPSI,PW,IND)
C YVETTE IS AN EXTERNAL SUBROUTINE REQUIRED BY SUBROUTINE DVOGER
C
C
C KANT=8
C
C SIZE IS THE CONVERGENT CRITERIA FOR SYSTEM SOLN.
C SIZE=.001
C
C DO 15 I=1,NUMSNP
C PV(I)=PSI(1,I)
C DPSI(I)=.0
15 CONTINUE
C DO 50 I=1,NUPBP
C J=IUPBP(I)
C PV(J)=0.
50 CONTINUE
C IF (NCOUNT .NE. 1) GO TO 05
C
C DO 27 MM=1,NUMSNP
C PSIIV(MM)=PV(MM)
C DO 28 I=1,NUMSNP
C BIGC(MM,I)=0.
28 CONTINUE
27 CONTINUE
C DO 210 L=1,NUMEL
C LL=ITYPE(L)
C IF(LL.NE. 0) GO TO 210
C N1=ELNOD(L,1)
C N2=ELNOD(L,2)
C N3=ELNOD(L,3)
C DO 86 J=1,3
C DO 90 N=1,3
C ALFA(N,J)=PSIIV(N1)*CM(L,N,J,1)+PSIIV(N2)*CM(L,N,J,2)
C +PSIIV(N3)*CM(L,N,J,3)
1 CONTINUE
90 CONTINUE
86 CONTINUE
C DO 150 K=1,3
C KK=ELNOD(L,K)

```

```

240
241
242
243
244
245
246
201
202
203
204
205
206
207
208
209
210
211
212
213
214

```





```

215      DO 280 I=1,3
216      II=ELMOD(L,I)
217
218      BIGC(KK,II)=BIGC(KK,II)+ALFA(K,II)*ZOMEGA(KK)
219      CCNTINUE
220      CCNTINUE
221      CCNTINUE
222
223      INSRRTION OF BOUNDARY POINTS
224
225      DO 285 I=1,NUPBP
226      II=IUPBP(I)
227      DO 281 J=1,NUMSNP
228      BIGC(J,II)=0.
229      BIGC(II,J)=0.
230      CCNTINUE
231      CCNTINUE
232      DO 311 I=1,NUMSNP
233      DO 310 J=1,NUMSNP
234      BIGCC(I,J)=0.
235      PART(I,J)=0.
236      DO 300 K=1,NUMSNP
237      BIGCC(I,J)=BIGCC(I,J)+(BIGAB(K,J)-BIGC(K,J))*BIGA(I,K)
238      PART(I,J)=PART(I,J)+(BIGAB(K,J)-2.*BIGC(K,J))*BIGA(I,K)
239      CCNTINUE
240      CCNTINUE
241      CCNTINUE
242
243      300
244      310
245      311
246
247      05 CONTINUE
248
249      IF (IND .EQ. 1) GO TO 200
250
251      DO 81 K=1,NUMSNP
252      DPT(K)=0.
253
254      DO 80 I=1,LCON
255      NAME=MNCD(K,I)
256      DPT(K)=PV(NAME)*BIGCC(K,I)+DPT(K)
257      CCNTINUE
258      80
259      81
260      CCNTINUE
261
262      IK=0
263      DO 120 IQ=1,NUMSNP
264      DO 100 K=1,NUMEQ
265      HOLD(K)=0.
266      DPSI(K)=0.
267
268
269

```







```

THIS IS THE LAST OF THE EXTRA FOR MTH EQ 1
CC C
END
SUBROUTINE CRANKO (NUMEL,NUMSNP,NUPBP,LCON,H,TQ,ERRVAL,
1 EPSVAL,IUPBP,PSIIV,ITYPE,ELNOD,CM,MNOD,ZOMEGA,BIGAB,BIGA,BIGA,
2 IPT1,IPT2,IPT3,PV,PSTEP,BIGD,BIGE,ASTR,ESTR,
3 BIGC,BIGCC,HOLD,TF)
CC C C
INTEGER*4 ELNOD
DIMENSION IUPBP(NUPBP),PSIIV(NUMSNP),ITYPE(NUMSNP),
1 ELNOD(NUMEL,3),CM(NUMEL,3,3,3),MNOD(NUMSNP,LCON),
2 ZOMEGA(NUMSNP),BIGAB(NUMSNP,LCON),BIGA(NUMSNP,LCON),
3 BATCH(400,4),PSTEP(NUMSNP),ALFA(3,3),PV(NUMSNP)
DIMENSION BIGD(NUMSNP,LCON),BIGE(NUMSNP,LCON),
1 ASTR(NUMSNP,LCON),ESTR(NUMSNP),BIGC(NUMSNP,LCON),
2 BIGCC(NUMSNP,LCON),HOLD(NUMSNP)
CC C C C C C C C C
DLT IS THE MAX TIME INTERVAL ATTEMPTED FOR ONE STEP.
EPSN IS THE CONVERGENCE CRITERIA FOR SOLN AT TIME T=T+DT
INITIALIZATION
DLT=H
PTIME=H
T=TO
LRGE = 1
EPSN=ERRVAL
STAR=EPSVAL
NIT=0
NUMEQ=NUMSNP-NUPBP
MIN=NUMSNP/4
CC C
NUF=NUMSNP/5
XNP=NUMSNP
XNUF=XNP/5.
IF ((XNUF-FLOAT(NUF)) .LT. 1.) NUF=NUF+1
C
DO 05 I=1,NUMSNP
PV(I)=.0
05 CONTINUE
CC C

```



```

C
C      BUILD THE BIGCC MATRIX
C
C 351 CONTINUE
C      DC 07 I=1,NUPBP
C      J=IUPBP(I)
C      PSIIV(J)=0.
C      CONTINUE
C 07 DO 09 I=1,NUMSNP
C      PSTEP(I)=PSIIV(I)
C      CONTINUE
C
C      DO 210 L=1,NUMEL
C      LL=ITYPE(L)
C      IF(LL.NE.0) GO TO 210
C      N1=ELNCD(L,1)
C      N2=ELNCD(L,2)
C      N3=ELNCD(L,3)
C      DO 88 J=1,3
C      DC 90 N=1,3
C      ALFA(N,J)=PSIIV(N1)*CM(L,N,J,1)+PSIIV(N2)*CM(L,N,J,2)
C      +PSIIV(N3)*CM(L,N,J,3)
C 1 CONTINUE
C 90 CONTINUE
C 88 DC 150 K=1,3
C      KK=ELNOD(L,K)
C      DO 180 I=1,3
C      II=ELNOD(L,I)
C      DC 91 MX=1,LCON
C      NCW=MX
C      MM=MNOD(KK,MX)
C      IF (MM.EQ.1) GO TO 92
C 91 CONTINUE
C 92 BIGC(KK,NOW)=BIGC(KK,NOW)+ALFA(K,I)*ZOMEGA(KK)
C      CONTINUE
C 180 CONTINUE
C 150 CONTINUE
C 210 CONTINUE
C      WRITE(6,755)
C 755 FORMAT(//10X,' BIGC MATRIX BEFORE BOUNDARY POINTS',//)
C      WRITE(6,778)((BIGC(I,J),J=1,LCON),I=1,NUMSNP)
C 778 FCORMAT(2X,8(E14.5))
C
C      INSERTION OF BOUNDARY POINTS
C      DC 185 I=1,NUPBP
C      II=IUPBP(I)
C      DC 191 MX=1,LCON

```





```

191 BIGC(II,MX)=-.0
185 CCNTINUE
C WRITE(6,757)
C FORMAT(///10X,'BIGC MATRIX WITH BOUNDARY POINTS',/)
C WRITE(6,778)((BIGC(I,J),J=1,LCON),I=1,NUMSNP)
C
C
C
310 DC 310 I=1,NUMSNP
C DO 310 J=1,LCON
C BIGCC(I,J)=BIGAB(I,J)-BIGC(I,J)
C CONTINUE
C WRITE(6,759)
C FORMAT(///10X,'BIGCC MATRIX',/)
C WRITE(6,778)((BIGCC(I,J),J=1,LCON),I=1,NUMSNP)
C
C
C
C BEGIN STEPING OUT IN TIME.
C
C STP = 4.
C ITRY=0
C
C COMPUTE PSI BASED ON A TRIAL DT.
C DO 55 JC=1,12
C
C FIRST COMPUTE (2/DT)*A=ASTR, ASTR-C=D, ASTR+D=E
C DS=2./DLT
C DO 15 KA=1,NUMSNP
C DO 13 KB=1,LCON
C ASTR(KA,KB)=DS*BIGA(KA,KB)
C BIGD(KA,KB)=ASTR(KA,KB)-BIGCC(KA,KB)
C BIGE(KA,KB)=ASTR(KA,KB)+BIGCC(KA,KB)
C CONTINUE
13 CONTINUE
C
C DO 19 KA=1,NUMSNP
C ESTR(KA)=0.
C DO 17 KB=1,LCON
C NAME = MNOD(KA,KB)
C ESTR(KA)=ESTR(KA)+BIGE(KA,KB)*PSIIV(NAME)
17 CONTINUE
19 CONTINUE
C
C THE SYSTEM IS NOW IS THE FAMILIAR FORM D(I,J)*PV(J)=ESIR(J)
C AND MAY BE SOLVED USING ANY TECHNIQUE FOR LINEAR SIMULTANEOUS
C EQNS. HERE GAUSS-SEIDEL ITERATION IS USED.
C
C ACCOMPLISH THE FIRST ITERATION BASED ON THE PV VALUE AT THE

```



```

C LAST SUCCESSFUL TIME POINT. SECOND AND SUCCESSIVE ITERATIONS
C ARE BASED ON UPDATED VALUED OF PV. A TEST FOR CONVERGENCE IS
C MADE, AND FAILURE RESULTS IN ANOTHER ATTEMPT WITH A SMALLER DT
C
C CREATION OF ESTR(I)=BIGE(I,J)*PSIIV(I)
C PERFORMED ONCE ON EACH ATTEMPT TO FIND A PROPER TIME INTERVAL.
C
DC 45 MTRY=1,NUMEQ
KT=0
DC 25 K=1,NUMEQ
HOLD(K)=0.
PV(K)=0.
DC 20 L=2,LCON
NAME = MNJD(K,L)
HOLD(K)= HOLD(K) + BIGD(K,L)*PSTEP(NAME)
20 CCNTINUE
PV(K)=(ESTR(K)-HOLD(K))/BIGD(K,1)
PDIF=(PV(K)-PSTEP(K))/PV(K)
24 CCNTINUE
IF (ABS(PDIF).LT. EPSN) KT =KT+1
PSTEP(K)=PV(K)
25 CCNTINUE
IF (KT .GE. NUMEQ) GO TO 65
45 CCNTINUE

C
C 291 FORMAT (/10X,'NO SOLN FOUND IN INTERVAL ',G10.4,' PLUS ',
1 G10.4)
WRITE(6,291) T,DLT
C
C PENALTY --- MAKE THE INTERVAL EVEN SMALLER.
DLT=DLT/STP
STP=STP+2.
ITRY=ITRY+1
55 CCNTINUE

C
C WRITE(6,292)
292 FORMAT(/10X,'A SOLN IS NOT POSSIBLE. THE PROGRAM SURRENDERS.')
```

```

C STOP
C
```

```

65 CCNTINUE
T=T + DLT
ITRA=MTRY+NUMEQ*ITRY
350 WRITE(6,350) T,DLT,ITRA
FORMAT (5X,'T= ',G12.6,4X,'DT= ',G12.6,4X,
```



```

1      'ITERATIONS REQUIRED = ',I3)
C
C      IF THE SOLN REQUIRED LESS ITERATIONS THAN PREVIOUSLY,
C      INCREASE THE SIZE OF THE INITIAL TIME STEP.
C      IF (ITRA .LT. LRGE .OR. ITRA .LT. MIN) CLT=CLT*.5
C      LRGE = ITRA

      IF ((T-PTIME)/PTIME .LT. STAR) GO TO 377
      PTIME=T
      WRITE (6,354)
      FORMAT(/(5(4X,'NODE',7X,'PSI',4X)))
354    DO 355 I=1,NUF
      I1=I+NUF
      I2=I+2*NUF
      I3=I+3*NUF
      I4=I+4*NUF
      WRITE(6,356) (I,PV(I),I1,PV(I1),I2,PV(I2),I3,PV(I3),I4,PV(I4))
355    CCNTINUE
356    FORMAT(5(4X,I3,3X,1PE12.4))
      WRITE(6,357)
357    FORMAT(/10X)
700    FORMAT(/1X,'VALUES OF IPT-1, IPT-2, IPT-3'/)
C

      NIT= 1 + NIT
      IF (NIT .GT. 400) GO TO 377
      BATCH (NIT,1)=T
      BATCH (NIT,2)= PV(IPT1)
      BATCH (NIT,3)= PV(IPT2)
      BATCH (NIT,4)= PV(IPT3)
377    CCNTINUE
C

      DO 27 MN=1,NUMSNP
      PSIV(MN)=PV(MN)
      DO 28 I=1,LCON
      BIGC(MN,I)=.0
      CCNTINUE
27    CONTINUE
      IF (T .LT. TF) GO TO 351
399    FORMAT(10X,I4)
      WRITE (6,399) NIT
      WRITE (3) BATCH
      WRITE (4) BATCH
      STCP
      END

```

















```

C      WRITE      (6,120)
120  FORMAT (1X, '//1X, 'ELEMENT', 5X, 'A1', 9X, 'A2', 9X, 'A3', 9X, 'B1', 9X, 'B2',
C      19X, 'B3', 8X, 'AREA'//)

C      DC 140 I=1, NUMEL
      A(I,1)=R3(I)-R2(I)
      A(I,2)=R1(I)-R3(I)
      A(I,3)=R2(I)-R1(I)
      B(I,1)=Z2(I)-Z3(I)
      B(I,2)=Z3(I)-Z1(I)
      B(I,3)=Z1(I)-Z2(I)
      AREA(I)=0.5*(A(I,2)*B(I,1)-A(I,1)*B(I,2))

C      WRITE(6,130) I, A(I,1), A(I,2), A(I,3), B(I,1), B(I,2), B(I,3),
1      AREA(I)
140  CONTINUE
130  FORMAT (3X, I3, 3X, 7(F12.7, 1X))

C      -----
C      ----- ZERO THE BIGA, BIGB, BIGC AND BIGAB MATRICES -----
C      -----
C      -----
C      DC 194 KK=1, NUMSNP
C      DC 193 II=1, NUMSNP
C      BIGA(KK, II)=0.
C      PART(KK, II)=0.
C      BIGAB(KK, II)=0.0
C      BIGC(KK, II)=0.
193  CONTINUE
194  CONTINUE

C      -----
C      ----- PEPSI CALCULATES THE BIGA MATRIX AND PART OF THE BIGAB -----
C      -----
C      -----
C      CALL PEPSI (NUMEL, NUMSNP, R1, R2, R3, AREA, ELNOD,
1      BIGA, BIGAB, VLMBDA)

C      WRITE(6,195)
195  FORMAT(//5X, 'BIGA MATRIX', //)
196  FORMAT(//5X, 'BIGAB MATRIX FROM PEPSI, NO DEL#2 CONTRIBUTION', //)
C

```



```

C-----
C CRUISE CALCULATES THE BIGAB MATRIX
C-----
C
C      CALL CRUISE (NUMEL,NUMSNP,      R1,R2,R3,AREA,ELNOD,
C      1      BIGAB,
C      A,B,VD)
C
C      WRITE(6,197)
C      FORMAT(/5X,'BIGAB MATRIX FROM CRUISE, WITH DEL**2 TERM',//)
C
C      DO 255 I=1,NUPBP
C      IF=IUPBP(I)
C      DO 251 J=1,NUMSNP
C      BIGAB(J,I)=0.
C      BIGAB(I,J)=0.
C      BIGAB(I,I)=0.
C      BIGAB(I,I,J)=0.
C      CCNTINUE
C      251 BIGAB(I,I,I)=1.
C      BIGAB(I,I,I)=0.
C      CCNTINUE
C      255
C
C      D1=1.
C      CALL LINV3F(BIGA,B,1,NUMSNP,NUMSNP,D1,D2,PART,IER)
C
C      WRITE(6,250)
C      FORMAT(/5X,'BIGA INVERSE',//)
C
C      WRITE OUT NUCLEAR DATA
C
C      WRITE(6,3105) ZNU,FISFAC,HBAR,EPSVAL,ERRVAL,AFUEL
C      3105 FORMAT(/5X,'NUCLEAR DATA',/2X,'ZNU=',G12.6/2X,'FISFAC=',G12.6/2X,
C      1      'HBAR=',G12.6/2X,'EPSVAL=',G12.6/2X,'ERRVAL=',G12.6/2X,'AFUEL=',
C      2      G12.6)
C      WRITE(6,3110)
C      3110 FORMAT(1H0.2X,'NODE',6X,'D',8X,'SGA',9X,'SGF',3X,'ALPHA',9X,'VD',
C      1      '7X,'ZLMBDA',7X,'VLMBDA',7X,'ZOMEGA',8X,'V',9X,'RHO($).')
C      DO 3115 I=1,NUMSNP
C      WRITE(6,3120) (I,D(I),SGA(I),SGF(I),ALPHA(I),VD(I),ZLMBDA(I),
C      1      'VLMBDA(I),ZOMEGA(I),V(I),RHO(I))
C      3115 CCNTINUE
C      3120 FORMAT(2X,I4,10(1PE12.4))
C
C      INTEGRATION BY DVOGER
C

```





```

EXTERNAL YVETTE
  CALL YVET( NUMEL, NUMEQ, NUMSNP, IUPBP, PSI IV, NUPBP, BIGC,
1  IYPE, ELNOD, CM, BIGA, BIGAB, BIGCC, PART, NWK, PSI, PW, DPSI,
2  PDPSI, HOLD, DPT, PV, NCOUNT, ZOMEGA, IPT1, IPT2, IPT3)
C----- INITIALIZE ARGUMENTS OF DVOGER -----
C
  PTIME= 1E-17
  STAR=.01
  NIT=0
  T=10
  JSTART=0
  EPS= EPSVAL
  IER=0
C
  WRITE(6,999) TITLE
999 FORMAT(1H1,20A4)
C
  WRITE(6,317)
317 WRITE(6,318) T, MTH, MAXDER, JSTART, H, HMIN, HMAX, EPS, NCOUNT
318 FORMAT(//10X, 'INITIAL ARGUMENTS',//)
12X, 'H=', G10.4, 2X, 'MTH=', I4, 2X, 'MAXDER=', I4, 2X, 'JSTART=', I4,
2, 'ACOUNT=', I2, /)
C
DO 320 I=1, NUMSNP
  PSI(1,I)=PSI IV(I)
320 CONTINUE
C
DO 321 I=1, NUPBP
  J=IUPBP(I)
  PSI(1,J)=0.0
321 CCNTINUE
C
  NUMEQ=NUMSNP-NUPBP
C
C
C
C
C-----
CCNSTRUCT THE BIGC MATRIX ON THE ELEMENT LEVEL
C
DO 200 L=1, NUMEL
DC 333 I=1,3
DC 332 J=1,3

```



```

DC 331 K=1,3
CM(L,I,J,K)=0.
331 CONTINUE
332 CONTINUE
333 CONTINUE
C
LL=ITYPE(L)
IF(LL.NE.0) GO TO. 200
CC=PI*AREA(L)/180.
C
CM(L,3,3,3)=CC*(6.0*R1(L)+4.0*R2(L)+2.4.0*R3(L))
CM(L,3,3,2)=CC*(4.0*R1(L)+2.0*R2(L)+6.0.0*R3(L))
CM(L,3,3,1)=CC*(2.0*R1(L)+4.0*R2(L)+6.0.0*R3(L))
CM(L,3,2,3)=CC*(2.0*R1(L)+4.0*R2(L)+6.0.0*R3(L))
CM(L,3,2,2)=CC*(2.0*R1(L)+2.0*R2(L)+2.0.0*R3(L))
CM(L,3,2,1)=CC*(4.0*R1(L)+2.0*R2(L)+6.0.0*R3(L))
CM(L,3,1,3)=CC*(6.0*R1(L)+4.0*R2(L)+2.0.0*R3(L))
CM(L,3,1,2)=CC*(2.0*R1(L)+2.0*R2(L)+4.0.0*R3(L))
CM(L,3,1,1)=CC*(2.0*R1(L)+6.0*R2(L)+4.0.0*R3(L))
CM(L,2,3,3)=CC*(2.0*R1(L)+6.0*R2(L)+4.0.0*R3(L))
CM(L,2,3,2)=CC*(2.0*R1(L)+2.0*R2(L)+2.0.0*R3(L))
CM(L,2,3,1)=CC*(4.0*R1(L)+2.0*R2(L)+6.0.0*R3(L))
CM(L,2,2,3)=CC*(6.0*R1(L)+4.0*R2(L)+2.0.0*R3(L))
CM(L,2,2,2)=CC*(2.0*R1(L)+2.0*R2(L)+4.0.0*R3(L))
CM(L,2,2,1)=CC*(2.0*R1(L)+6.0*R2(L)+4.0.0*R3(L))
CM(L,2,1,3)=CC*(4.0*R1(L)+2.0*R2(L)+6.0.0*R3(L))
CM(L,2,1,2)=CC*(2.0*R1(L)+4.0*R2(L)+2.0.0*R3(L))
CM(L,2,1,1)=CC*(6.0*R1(L)+2.0*R2(L)+4.0.0*R3(L))
CM(L,1,3,3)=CC*(4.0*R1(L)+2.0*R2(L)+6.0.0*R3(L))
CM(L,1,3,2)=CC*(2.0*R1(L)+4.0*R2(L)+2.0.0*R3(L))
CM(L,1,3,1)=CC*(6.0*R1(L)+2.0*R2(L)+4.0.0*R3(L))
CM(L,1,2,3)=CC*(6.0*R1(L)+4.0*R2(L)+2.0.0*R3(L))
CM(L,1,2,2)=CC*(2.0*R1(L)+2.0*R2(L)+4.0.0*R3(L))
CM(L,1,2,1)=CC*(4.0*R1(L)+6.0*R2(L)+2.0.0*R3(L))
CM(L,1,1,3)=CC*(2.0*R1(L)+6.0*R2(L)+4.0.0*R3(L))
CM(L,1,1,2)=CC*(2.0*R1(L)+2.0*R2(L)+2.0.0*R3(L))
CM(L,1,1,1)=CC*(4.0*R1(L)+4.0*R2(L)+6.0.0*R3(L))
C 200 CCNTINUE
C 351 CONTINUE
IF(NCOUNT.EQ.1)GO TO 312
C
DO 210 L=1,NUMEL
LL=ITYPE(L)
IF(LL.NE.0) GO TO 210
N1=ELNOD(L,1)

```







```

IF (IER.EQ.0) GO TO 3540
JSTART=-1
H=HMIN*.1
HMIN=H*.1
GO TO 350
3540 CONTINUE
JSTART=1
JF((T-PTIME)/PTIME .LT. STAR) GO TO 362
PTIME=T
WRITE(6,3510)T,H,JSTART,IER
3510 FORMAT(/2X,'T=',G12.6,2X,'H=',G12.6,2X,'JSTART=',I4,2X,'IER=',I4)
C
354 DO 355 I=1,NUF
WRITE(6,354)
354 FORMAT(/5(4X,'NODE',7X,'PSI',4X)))
I1=1+NUF
I2=I+2*NUF
I3=I+3*NUF
I4=I+4*NUF
WRITE(6,356) (I,PSI(I,1),I1,PSI(I,1,I1),I2,PSI(I,1,I2),I3,PSI(I,1,I3),I4,
1,PSI(I,1,I4))
355 CONTINUE
356 FORMAT(5(4X,I3,3X,1PE12.4))
C
NIT=1+NIT
IF (NIT.GT.400) GO TO 377
BATCH (NIT,1)=T
BATCH (NIT,2)=PSI(1,IPT1)
BATCH (NIT,3)=PSI(1,IPT2)
BATCH (NIT,4)=PSI(1,IPT3)
377 CONTINUE
C
DC 360 I=1,NUMSNP
R(I)=PSI(2,I)/H
360 CONTINUE
C
357 WRITE(6,357)
357 FORMAT(/5(4X,'NODE',7X,'DPSI',3X)))
DO 359 I=1,NUF
I1=I+NUF
I2=I+2*NUF
I3=I+3*NUF
I4=I+4*NUF
WRITE(6,356) (I,R(I),I1,R(I1),I2,R(I2),I3,R(I3),I4,R(I4))
359 CONTINUE
362 CONTINUE
C

```









```

AMATRIX(2,1)=AMATRIX(1,2)
AMATRIX(2,2)=COEFFFA *(2.0*R1(L)+6.0*R2(L)+2.0*R3(L))
AMATRIX(2,3)=COEFFFA *(R1(L)+2.0*R2(L)+2.0*R3(L))
AMATRIX(3,1)=AMATRIX(1,3)
AMATRIX(3,2)=AMATRIX(2,3)
AMATRIX(3,3)=COEFFFA *(2.0*R1(L)+2.0*R2(L)+6.0*R3(L))

```

```

-----
C STORE ELEMENT MATRIX, AMATRIX, INTO SYSTEM MATRIX, BIGA
C-----

```

```

DO 20 K=1,3
KK=ELNOD(L,K)
DO 10 I=1,3
II=ELNOD(L,I)
BIGA(KK,II)=BIGA(KK,II)+AMATRIX(K,I)
BIGAB(KK,II)=BIGAB(KK,II)+VLMBDA(KK)*AMATRIX(K,I)
10 CONTINUE
20 CONTINUE

```

```

C 200 CONTINUE

```

```

RETURN

```

```

END

```

```

SUBROUTINE CRUISE (NUMEL,NUMSNP, R1,R2,R3,AREA,ELNOD,BIGAB,
A,B,VD)

```

```

*****
CRUISE CALCULATES THE BIGAB MATRIX
*****

```

```

INTEGER*4 ELNOD

```

```

1 DIMENSION R1(NUMEL),R2(NUMEL),R3(NUMEL),AREA(NUMEL),
2 BIGAB(NUMSNP,NUMSNP),BMATRIX(3,3),
ELNOD(NUMEL,3),A(NUMEL,3),B(NUMEL,3),VD(NUMSNP)

```

```

PI=3.1415927

```

```

C CALCULATE THE 3X3 B(I,J) MATRIX FOR ELEMENT L
C-----

```

```

DO 200 L=1,NUMEL

```



```

C      COEFFB= PI / (6.0 * AREA(L) )
C      BMATRIX(1,1)=COEFFB *(-R1(L))*(2.0*B(L,1)**2+2.0*A(L,1)**2)-R2(L)*
1A(L,1)+B(L,2)+B(L,1)**2+A(L,1)*A(L,2)+A(L,1)**2)-R3(L)*(B(L,1)*
2B(L,3)+B(L,1)**2+A(L,1)*A(L,3)+A(L,1)**2)+2.0*AREA(L)*B(L,1))
C      BMATRIX(1,2)=COEFFB *(-R1(L))*(2.0*B(L,1)*B(L,2)+2.0*A(L,1)*
1A(L,2))-R2(L)*(B(L,2)*B(L,1)+A(L,2)*A(L,1)**2+A(L,1)*A(L,2))-
2R3(L)*(B(L,2)*B(L,3)+B(L,1)*B(L,2)+A(L,2)*A(L,1)*A(L,2))+
32.0*AREA(L)*B(L,2))
C      BMATRIX(1,3)=COEFFB *(-R1(L))*(2.0*B(L,1)*B(L,3)+2.0*A(L,1)*
1A(L,3))-R2(L)*(B(L,3)+B(L,1)*B(L,3)+A(L,2)*A(L,3)+A(L,1)*
2A(L,3))-R3(L)*(B(L,3)**2+B(L,1)*B(L,3)+A(L,3)**2+A(L,1)*A(L,3))+
32.0*AREA(L)*B(L,3))
C      BMATRIX(2,1)=BMATRIX(1,2)
C      BMATRIX(2,2)=COEFFB *(-R1(L))*(B(L,2)**2+B(L,1)*B(L,2)+A(L,2)**2+
1A(L,1)*A(L,2))-R2(L)*(2.0*B(L,2)**2+2.0*A(L,2)*A(L,2))-R3(L)*(B(L,3)*
2B(L,2)+B(L,2)**2+A(L,3)*A(L,2)+A(L,2)**2)+2.0*AREA(L)*B(L,2))
C      BMATRIX(2,3)=COEFFB *(-R1(L))*(B(L,2)*B(L,3)+B(L,1)*B(L,3)+A(L,2)*
1A(L,3)+A(L,1)*A(L,3))-R2(L)*(2.0*B(L,2)*B(L,3)+2.0*A(L,2)*A(L,3))-
2R3(L)*(B(L,3)**2+B(L,2)*B(L,3)+A(L,3)**2+A(L,2)*A(L,3))+
32.0*AREA(L)*B(L,3))
C      BMATRIX(3,1)=BMATRIX(1,3)
C      BMATRIX(3,2)=BMATRIX(2,3)
C      BMATRIX(3,3)=COEFFB *(-R1(L))*(B(L,3)**2+B(L,1)*B(L,3)+A(L,3)**2+
1A(L,1)*A(L,3))-R2(L)*(B(L,3)**2+B(L,2)*B(L,3)+A(L,3)**2+A(L,2)*
2A(L,3))-R3(L)*(2.0*B(L,3)**2+2.0*A(L,3)*A(L,3)+2.0*AREA(L)*B(L,3))
C
C
C      STORE ELEMENT MATRIX, AMATRIX, INTO SYSTEM MATRIX, BIGA
C
C      DC 20 K=1,3
C      KK=ELNOD(L,K)
C      DO 10 I=1,3
C      II=ELNOD(L,I)
C      BIGAB(KK,II)=BIGAB(KK,II)+VD(KK)*BMATRIX(K,I)
C
C

```

```

10 CONTINUE
20 CCNTINUE

```



```

C 200 CONTINUE
C
C RETURN
C
C END
C SUBROUTINE YVET(NUMEL,NUMEQ,NUMSNP, IUPBP,PSIIV,NUPBP,BIGC,
C ITYPE,ELNOD,CM,BIGA,BIGAB, BIGCC,PART,NWK,PSI,PW,DPSI,
C PDPSI,HOLD,DPT,PV,NCOUNT,ZOMEGA,IPT1,IPT2,IPT3)
C
C *****
C YVETTE IS AN EXTERNAL SUBROUTINE REQUIRED BY SUBROUTINE CVOGER
C *****
C
C INTEGER*4 ELNOD
C
C DIMENSION IUPBP(NUPBP),PSIIV(NUMSNP),BIGC(NUMSNP,NUMSNP),
C ELNOD(NUMEL,3),CM(NUMEL,3,3),BIGA(NUMSNP,NUMSNP),
C BIGAB(NUMSNP,NUMSNP),BIGCC(NUMSNP,NUMSNP),
C PART(NUMSNP,NUMSNP),PDPSI(NUMSNP),PV(NUMSNP),PSI(8,NUMSNP),
C ALFA(3,3),DPSI(NUMSNP),PW(NWK),ITYPE(NUMEL)
C
C PI=3.1415927
C
C 55 FORMAT (2(3X,1PE12.4))
C WRITE(6,55) BIGA(1,1),BIGAB(1,1)
C RETURN
C
C ENTRY YVETTE(PSI,T,M,DPSI,PW,IND)
C
C DO 15 I=1,NUMSNP
C PV(I)=PSI(1,I)
C 15 CONTINUE
C DO 50 I=1,NUPBP
C J=IUPBP(I)
C PV(J)=0.
C 50 CONTINUE
C
C IF (NCOUNT .NE. 1) GO TO 05
C
C DO 27 MM=1,NUMSNP
C PSIIV(MM)=PV(MM)
C DO 28 I=1,NUMSNP
C BIGC(MM,I)=0.

```

240  
241  
242  
243  
244









```

C      DPSI(K)=0.
C      IF(K.GT.NUML) GO TO 100
C
C      DO 80 I=1,NUMSNP
C      DPSI(K)=DPSI(K)+BIGCC(K,I)*PV(I)
C      CCNTINUE
C      80
C      100 CCNTINUE
C
C      IF(T.GT.0.0) RETURN
C      WRITE(6,60)T
C      60 FORMAT(//2X,'FROM YVETTE',5X,'T=',G20.10)
C      WRITE(6,65)(DPSI(K),K=1,NUMSNP)
C      65 WRITE(6,65)(PSI(I,K),K=1,NUMSNP)
C      FCFORMAT(10(IX,E11.5))
C      RETURN
C
C      205 CCNTINUE
C      DO 900 K=1,NUMSNP
C      DO 880 L=1,NUMSNP
C      KL=NUMSNP*(L-1)+K
C      PW(KL)=PART(K,L)
C      880 CCNTINUE
C      900 CCNTINUE
C
C      IF(T.GT.0.0) GO TO 950
C      WRITE(6,65) (PW(I),I=1,1444)
C      RETURN
C
C      950 WRITE(6,700)
C      700 FORMAT(//IX,'VALUES OF PW(IPT1),PW(IPT2),PW(IPT3)://)
C      710 WRITE(6,710) PW(IPT1),PW(IPT2),PW(IPT3)
C      FCFORMAT(5X,3G20.10)
C      RETURN
C      END

```



# DATA PROCESSOR FOR FED2 PROGRAMS.

THIS PROGRAM TAKES THE STORED DATA SETS AND PRESENTS A DIFFERENT METHOD AND/OR DISTURBANCE. THE METHODS MUST BE IN THE ORDER GIVEN. ALL METHODS FOR ONE DISTURBANCE ARE RUN TOGETHER. THE ORDER FOR DISTURBANCE TYPES IS 1-CENTRAL, 2-UNFIROM, 3-SKEW. THE PROGRAM IS EASILY MODIFIED TO ACCEPT CHANGES, EG. THE EXCLUSION OF A DATA SET OR METHOD, OR THE EXAMINATION OF A DIFFERENT TIME INTERVAL.

## THE METHODS ARE

```

EXACT          N**3
LINEARIZED     N*N
CRANK-NICOLSON
GEAR           COMPACTED
ADAMS          COMPACTED

```

## EXPLANATION OF ITEMS.

NIT IS THE NUMBER OF TIME POINTS CONTAINED IN A GIVEN DSET. EACH TIME A DSET IS READ IN, ITS CORRESPONDING NIT MUST BE READ IN FROM A CARD.

JJ IS THE NUMBER OF DIFFERENT DISTURBANCES EXAMINED.

JTYPE IS THE NUMBER OF DIFFERENT METHODS.

TCUT IS THE LAST NORMALIZED TIME POINT.

TSTRT IS THE APPROXIMATE FIRST NORMALIZED TIME POINT.

NUM IS THE NUMBER OF NORMALIZED TIME POINTS.

## HOUSKEEPING.

```

DIMENSION FLUX(200),FTIM(200)
DIMENSION STUF(50),T(50),HOLD(400,4),DSET(400,4)
DIMENSION EXACT(50,3),XLIN(50,3),ADAM(50,3),
1 CRNK(50,3),GEER(50,3)
DIMENSION DIFX(50,3),DIFA(50,3),DIFG(50,3),DIFC(50,3)
DIMENSION P(50)
DIMENSION H(1800)
DIMENSION STOR(50),ITB(12),RTB(28)

DATA ITB,RTB/12*0,28*.0/
DATA CTR,UNF,SKW/'CENR','UNIF','SKEW'/
DATA CTRL,CORE,REFL/'CTRL','CORE','REFL'/
DATA CR/'CRNK'/
DATA GE/'GEAR'/
DATA AD/'ADAM'/
DATA EX/'EXCT'/
DATA XL/'XLIN'/
DATA PT/'.....'/

```

```

110 FORMAT (5X,I5)
205 FORMAT ('1',10X,'SINGLE CENTRAL DISTURBANCE'//)
207 FORMAT ('1',10X,'UNIFORM DISTURBANCE'//)
209 FORMAT ('1',10X,'SKEW DISTURBANCE'//)
211 FORMAT (10X,'EXACT'//)
212 FORMAT (10X,'XLIN'//)
213 FORMAT (10X,'ADAM'//)

```



```

214 FORMAT (10X,'GEAR'//)
215 FORMAT (10X,'CRNK'//)
222 FORMAT(2X,'TIME',10X,'PSI(1)',8X,'PSI(13)',9X,
1 'PSI(25)',8X,
2 'TIME',10X,'PSI(1)',8X,'PSI(13)',9X,'PSI(25)'//)
224 FORMAT ( 2X,4(2X,1PE12.4))
232 FORMAT (/10X,'PSI(1) CENTER POINT'//)
234 FORMAT (/10X,'PSI(13) CORE POINT'//)
236 FORMAT (/10X,'PSI(25) REFLECTOR POINT'//)
238 FORMAT (/10X,'VALUES COMPARED IN PERCENT DIFFERENCE.')
240 FORMAT (/10X,'NORMALIZED TIME VALUES'//)
242 FORMAT (10X,'TIME',10X,'EXACT',10X,'LINEARIZED',10X,
1 'GEER',10X,'ADAMS',8X,'CRANK-N'//)
244 FORMAT ((7X,6(4X,E11.4)))
246 FORMAT (7X,E11.4,4X,E11.4,08X,F6.2,10X,F6.2,9X,F6.2,
1 8X,F6.6)
500 FORMAT ('1'//10X,' ')

```

# ESTABLISH THE CONTROLLING PARAMETERS.

```

SCALE =1.E 14
TSTRT=1.E-5
NUM=50
VAL=.05
JTYPE=3
JJ=3

```

# ZERO THE STORAGE AREAS.

```

DO 03 I=1,NUM
DO 04 J=1,3
EXACT (I,J)=.0
XLIN (I,J)=.0
ACAM (I,J)=.0
GEER (I,J)=.0
CRNK (I,J)=.0
DIFX (I,J)=.0
DIFA (I,J)=.0
DIFG (I,J)=.0
DIFC (I,J)=.0
04 CONTINUE
03 CONTINUE

```

```

DC 88 LL=1,JJ
LL DENOTES WHICH TYPE OF DISTURBANCE IS BEING
CONSIDERED.

```

# CONSTRUCT THE NORMALIZED TIME INCREMENTS.

```

XNUM=NUM
TCUT=.1
IF (LL .EQ. 2) TCUT=.0329
X=(TCUT/TSTRT)**(1./XNUM)
DO 21 I=1,NUM
T(I)=TSTRT*(X**I)
P(I)= ALG10(T(I))
21 CONTINUE

```

```

DO 86 LM=1,JTYPE

```

# READ AND WRITE ONE DATA SET IN THE RAW.

```

READ (5,110) NIT
READ (4) DSET
IF (LL .EQ. 1) WRITE(6,205)
IF (LL .EQ. 2) WRITE(6,207)
IF (LL .EQ. 3) WRITE(6,209)
IF (LM .EQ. 1) WRITE(6,211)

```





```

C      IF (LM      .EQ. 2) WRITE(6,212)
      IF (LM      .EQ. 3) WRITE(6,213)
      IF (LM      .EQ. 3) WRITE(6,215)
      IF (LM      .EQ. 4) WRITE(6,214)
      IF (LM      .EQ. 5) WRITE(6,215)
      WRITE (6,224) ((DSET(I,K),K=1,4),I=1,NIT)

C
C      REBUILD THE DATA SET EXCLUDING TIME VALUES WHICH
C      ARE IDENTICAL.
C
      DO 06 I=1,4
      HOLD(1,I)= DSET(1,I)
06  CONTINUE

C      JK=1
      DO 07 I=2,NIT
      J=JK
      IF ((DSET(I,1)-HOLD(J,1))/HOLD(J,1) .LT. VAL) GO TO 07
      JK=JK+1
      DO 05 JIM=1,4
      HOLD(JK,JIM)=DSET(I,JIM)
05  CONTINUE
07  CONTINUE

C
C      DO 17 M=2,4
C
C      SCREEN TO EXCLUDE DATA POINTS GOING NEGATIVE.
      DO 23 JEND=1,JK
      JDIF=JK-JEND + 1
      JP=JDIF+1
      IF (HOLD(JDIF,M) .LT. .0) GO TO 25
23  CONTINUE
      JP=JDIF
      GO TO 26
25  CONTINUE
      DO 26 I=1,JP
      SMOOTH OUT ANY DISCONTINUITY.
      HOLD(I,M)=HOLD(JP,M)
26  CONTINUE

C      DO 27 I=1,NUM
      STUF(I)=P(I)
27  CONTINUE

C
C      INSURE THE LAST TIME VALUE FOR DSET IS BEYOND THE
C      LAST NORMALIZED TIME POINT.
C
      JB=0
      DO 15 JF=1,JK
      JA=JK-JB
      JB=JB+1
      IF (HOLD(JA,1) .GE. T(NUM)) GO TO 19
      DO 13 I=2,4
      HOLD(JA,I)=SCALE
13  CONTINUE
15  CONTINUE
19  CONTINUE

C
C      SCALE THE INPUT VALUES FOR THE INTERPOLATION ROUTIN
C
      DO 30 I= 1,JK
      FTIM(I)=ALOG10(HOLD(I,1))
      FLUX(I)=      (HOLD(I,M))/SCALE
30  CONTINUE
      WRITE (6,224) ((HOLD(I,K),K=1,4),I=1,JK)

```



```

C      NORMALIZE THE DATA SET
C
JER=0
JT=JK
CALL ICS1VU (FLUX,FTIM,JT,NUM,STUF,H,JER)

C
C      TRANSFER NORMALIZED DSET TO APPROPRIATE CLASS AND
C      COMPUTE THE DIFFERENCE.
C
L=M-1
C      L ESTABLISHES WHICH POINT IN THE REACTOR IS BEING
C      EXAMINED.
GO TO (41, 43, 45, 47, 49),LM
41 CONTINUE
DO 42 I=1,NUM
EXACT(I,L)=STUF(I)
42 CONTINUE
GO TO 51
43 CONTINUE
DO 44 I=1,NUM
DIFX(I,L)=((STUF(I) -EXACT(I,L))/EXACT(I,L))*100.
XLIN(I,L)=STUF(I) *SCALE
44 CONTINUE
GO TO 51
45 CONTINUE
DO 46 I=1,NUM
CRNK(I,L)=STUF(I) *SCALE
DIFC(I,L)=((STUF(I) -EXACT(I,L))/EXACT(I,L))*100.
46 CONTINUE
GO TO 51
47 CONTINUE
DO 48 I=1,NUM
DIFG(I,L)=((STUF(I) -EXACT(I,L))/EXACT(I,L))*100.
GEER(I,L)=STUF(I) *SCALE
48 CONTINUE
GO TO 51
49 CONTINUE
DO 50 I=1,NUM
DIFA(I,L)=((STUF(I) -EXACT(I,L))/EXACT(I,L))*100.
ADAM(I,L)=STUF(I) *SCALE
50 CONTINUE
51 CONTINUE
17 CONTINUE
86 CONTINUE

C
DO 63 L=1,3
DO 61 I=1,NUM
EXACT(I,L)=EXACT(I,L)*SCALE
61 CONTINUE
63 CONTINUE

C
C
C      WRITE OUT RESULTS AND DRAW THE GRAPHS.
DO 57 LCNT=1,4
DO 58 L=1,3
IF (LL .EQ. 1) WRITE(6,205)
IF (LL .EQ. 2) WRITE(6,207)
IF (LL .EQ. 3) WRITE(6,209)

C
WRITE (6,240)
IF(L .EQ. 1) WRITE(6,232)
IF(L .EQ. 2) WRITE(6,234)
IF(L .EQ. 3) WRITE(6,236)
IF (LCNT .EQ. 2) WRITE(6,238)

C
WRITE(6,242)
IF (LCNT .EQ.1) WRITE(6,244)
1   (T(I),EXACT(I,L),XLIN(I,L),GEER(I,L),ADAM(I,L),CRNK
2   I=1,NUM)

```



```

      IF(LCNT .EQ. 2) WRITE(6,246)
1      (T(I),EXACT(I,L),DIFX(I,L),DIFG(I,L),DIFA(I,L),DIFC
2      I=1,NUM)
      LPLT=LPLT+1
      IF (LCNT .LE. 2) GO TO 54

```

```

C
C
C      INSERT CALLS TO PLOTTING ROUTINES.

```

```

      RTB(6)=PT
      RTB(2)=.0
      IF (LCNT .EQ. 4) RTB(2)=5.
      IF (LL .EQ. 1) RTB(5) =CTR
      IF (LL .EQ. 2) RTB(5) =UNF
      IF (LL .EQ. 3) RTB(5) = SKW
      IF (L .EQ. 1) RTB(7) =CTRL
      IF (L .EQ. 2) RTB(7) =CORE
      IF (L .EQ. 3) RTB(7) =REFL

```

```

C
      DC 97 IPLOT=1,JTYPE
      GO TO (71,73,75,77,79),IPLOT
71  CONTINUE
      RTB(3) =EX
      ITB(1) =1
      DO 72 I=1,NUM
      STUF(I) = EXACT(I,L)
      IF (LCNT .EQ. 4) STUF(I)=0.
72  CONTINUE
      GO TO 95
73  CONTINUE
      RTB(3) =XL
      ITB(1) = 2
      DO 74 I=1,NUM
      STUF(I)=XLIN(I,L)
      IF (LCNT .EQ. 4) STUF(I)=DIFX(I,L)
74  CONTINUE
      GO TO 95
75  CCNTINUE
      RTB(3) = CR
      ITB(1)=3
      DO 76 I=1,NUM
      STUF(I)=CRNK(I,L)
      IF (LCNT .EQ. 4) STUF(I)=DIFC(I,L)
76  CONTINUE
      GC TO 95
77  CONTINUE
      RTB(3) = GE
      DO 78 I=1,NUM
      STUF(I)=GEER(I,L)
      IF (LCNT .EQ. 4) STUF(I)=DIFG(I,L)
78  CONTINUE
      GO TO 95
79  CONTINUE
      RTB(3) = AD
      DO 80 I=1,NUM
      STUF(I) = ADAM(I,L)
      IF (LCNT .EQ. 4) STUF(I)=DIFA(I,L)
80  CCNTINUE
95  CCNTINUE
      CALL DRAWP(NUM,P,STUF,ITB,RTB)
97  CONTINUE

```

```

C
54  CCNTINUE
56  CCNTINUE
57  CCNTINUE

```

```

C
88  CONTINUE

```

```

      WRITE (6,500)
      STOP
      END

```



A PROGRAM TO TEST THE COMPACT METHOD  
OF FINITE ELEMENT MATRIX WORK.

INPUT INFORMATION

DATA CARDS ARE REQUIRED IN THE FOLLOWING ORDER,

NODY, NUMSNP (NR NODES IN Y DIRECTION, NR SYSTEM  
NCDAL POINTS)

PSI THE SET OF VALUES FOR TEST MULTIPLICATION.

NUMEL NR ELEMENTS IN THE MESH

ELNOD THE ELEMENT CONNECTIVITY MATRIX.

```

      INTEGER*4 ELNOD
      DIMENSION VALUE(50),MNOD(50,7),VMATX(50,7),
1      BIGA(50,50),ELNOD(75,3)
      DIMENSION PSI(50),COMPCT(50),USUAL(50)
      DIMENSION AA(3,3)
110  FORMAT (115,5X,15)
119  FORMAT ('1'////////// ' ')
120  FORMAT (F9.2,6F10.2)
150  FORMAT (5X,3I10)

210  FORMAT ('1',10X,'NODE NEIGHBOR CONNECTIVITY'//)
212  FORMAT (10X,'NODE',5X,'K2',5X,'K3',5X,'K4',5X,'K5',
1      5X,'K6',5X,'K7'//)
215  FORMAT (10X,7(I5,2X)//)
220  FORMAT ('1'////10X,'THIS IS THE NUMSNP X 7 MATRIX'//)
225  FORMAT (10X,7(F6.0,1X)//)
230  FORMAT ('1'////10X,'THE ELEMENT CONNECTIVITY MATRIX'//)
235  FORMAT (10X,14,3(3X,14))
250  FORMAT ('1'////10X,'THIS IS THE USUAL NUMSNP X NUMSNP ',
1      'BIGA MATRIX'//)
255  FORMAT (2X,20(F5.0,1X)//)
270  FORMAT ('1'////10X,'COMPACT FORM',5X,'USUAL FORM'//)
272  FORMAT (5X,12,7X,F8.0,07X,F8.0)

```

```

      READ(5,110) NODY,NUMSNP
      READ (5,120) (PSI (I),I=1,NUMSNP)
      WRITE (6,119)
      WRITE (6,120) (PSI (I), I=1,NUMSNP)

```

CONSTRUCT THE NEIGHBOR NODES OF THE HEXAGON.

```

      WRITE (6,210)
      WRITE (6,212)

```

```

      DO 35 K1=1,NUMSNP
      N=1
5  CONTINUE
      NN=N*NODY
      N=N+1
      IF ((K1-NN) .GT. 0) GO TO 5
      K3=K1-1
      K6=K1+1
      K2=K1-NCDY
      K7=K2+1
      K5=K1+NODY
      K4=K5-1
      IF(K1 .GT. NODY) GO TO 6
      K2=0
      K7=0

```





```

6 IF(K1 .NE. NN) GO TO 7
  K6=0
  K7=0
7 IF(K1 .LE. (NUMSNP-NODY)) GO TO 8
  K5=0
  K4=0
8 IF (K1 .NE. (NN-NODY+1)) GO TO 9
  K3=0
  K4=0
9 CONTINUE

```

C

```

  MNOD(K1,1)=K1
  MNOD(K1,2)=K2
  MNOD(K1,3)=K3
  MNOD(K1,4)=K4
  MNOD(K1,5)=K5
  MNOD(K1,6)=K6
  MNOD(K1,7)=K7
  WRITE (6,215) (MNOD(K1,I),I=1,7)
35 CCNTINUE

```

C  
C  
C  
C

```

  READ THE CONNECTIVITY MATRIX
  READ (5,110) NUMEL
  WRITE (6,230)
  DO 60 I=1,NUMEL
  READ (5,150) (ELNOD(I,J),J=1,3)
  WRITE (6,235) I,(ELNOD(I,J),J=1,3)
60 CONTINUE

```

C  
C

```

  ZERO THE BIGA, COMPCT, AND VALUE MATRICIES.
  DO 62 I=1,NUMSNP
  VALUE(I)=0.
  DO 61 J=1,NUMSNP
  VMATX(I,J)=0.
  BIGA(I,J)=0.
61 CONTINUE
62 CONTINUE

```

C  
C  
C  
C  
C  
C

CCNSTRUCT THE N X 7 MATRIX

```

DO 16 L=1,NUMEL
Y=FLOAT(L)
R1=2.*Y
R2=3.*Y
R3=4.*Y
AA(1,1)=2.*R1+R2+2.*R3
AA(1,2)=R1+R2+2.*R3
AA(1,3)=2.*R1+2.*R2+R3
AA(2,1)=2.*R1+2.*R2+2.*R3
AA(2,2)=R1+R2+R3
AA(2,3)=R1+2.*R2+2.*R3
AA(3,1)=AA(1,1)
AA(3,3)=AA(2,3)
AA(3,2)=AA(2,1)
DO 15 K=1,3
KK=ELNOD(L,K)
DO 14 I=1,3
II=ELNOD(L,I)

```

C  
C  
C  
C

```

  SEARCH KK ROW FOR NODE II. IF FOUND, POST THE VALUE
  TO VMATX(KK,M). OTHERWISE, CONTINUE STEPPING
  THROUGH THE ELEMENT.
DO 12 M=1,7
NOW=M
MM =MNOD(KK,M)
IF (MM .EQ. II) GO TO 13
12 CONTINUE

```



```

13 CONTINUE
  VMATX(KK,NOW)=VMATX(KK,NOW) + AA(K,I)
14 CONTINUE
15 CONTINUE
16 CCNTINUE

```

```

C
C
C      WRITE OUT THE N X 7
      WRITE (6,220)
      WRITE (6,212)
      DO 37 K1=1,NUMSNP
      WRITE (6,225) (VMATX(K1,J),J=1,7)
37 CONTINUE

```

```

C
C
C
C      GENERATE THE ELEMENT VALUES AND MOVE THEM INTO THE NXN

```

```

      DO 68 L=1,NUMEL
      Y=FLOAT(L)
      R1=2.*Y
      R2=3.*Y
      R3=4.*Y
      AA(1,1)=2.*R1+R2+2.*R3
      AA(1,2)=R1+R2+2.*R3
      AA(1,3)=2.*R1+2.*R2+R3
      AA(2,1)=2.*R1+2.*R2+2.*R3
      AA(2,2)=R1+R2+R3
      AA(2,3)=R1+2.*R2+2.*R3
      AA(3,1)=AA(1,1)
      AA(3,2)=AA(2,1)
      AA(3,3)=AA(2,3)
      DO 67 K=1,3
      KK= ELNOD(L,K)
      DO 66 I=1,3
      II=ELNOD(L,I)
      BIGA(KK,II)=BIGA(KK,II)+AA(K,I)
66 CONTINUE
67 CCNTINUE
68 CONTINUE

```

```

C
C
C      WRITE OUT THE N X N.
      WRITE (6,250)
      DO 75 I=1,NUMSNP
      WRITE (6,255) (BIGA(I,J), J=1,NUMSNP)
75 CONTINUE

```

```

C
C
C
C      PERFORM A SIMPLE MATRIX X COLM MULTIPLICATION ON BOTH
      AND COMPARE THE RESULTS.

```

```

      DO 72 I=1,NUMSNP
      CCMPCT(I)=0.
      DO 71 J=1,7
      NAME=MNOD(I,J)
      COMPCT(I)=PSI(NAME)*VMATX(I,J)+ COMPCT(I)
71 CCNTINUE
72 CCNTINUE

```

```

C
      DO 42 I=1,NUMSNP
      USUAL(I)=0.
      DO 41 J=1,NUMSNP
      USUAL(I)=BIGA(I,J)*PSI(J)+USUAL(I)
41 CONTINUE
42 CONTINUE

```

```

C
C
      WRITE(6,270)
      DO 80 I=1,NUMSNP
      WRITE(6,272) I, COMPCT(I),USUAL(I)
80 CONTINUE

```



STOP							
END							
//GO.SY	SIN	DD	*				
8.	10.		4	20	30.	40.	50.
85.	15.			25.	35.	45.	55.
87.	17.			27.	37.	47.	57.
	24						6
1	1			5		2	0.
2	5			6		2	5.
3	6			3		2	
4	6			7		3	
5	7			4		3	
6	7			8		4	
7	9			6		5	
8	9			10		6	
9	10			7		6	
10	10			11		7	
11	11			8		7	
12	11			12		8	
13	13			10		9	
14	13			14		10	
15	14			11		10	
16	14			15		11	
17	15			12		11	
18	15			16		12	
19	17			14		13	
20	17			18		14	
21	18			15		14	
22	18			19		15	
23	19			16		15	
24	19			20		16	



# COMPACT FORM

# USUAL FORM

1	780.	780.
2	8130.	8130.
3	21120.	21120.
4	25810.	25810.
5	15799.	15799.
6	54965.	54965.
7	98365.	98365.
8	72195.	72195.
9	39740.	39740.
10	126275.	126275.
11	195386.	195386.
12	113725.	113725.
13	92605.	92605.
14	241556.	241556.
15	288460.	288460.
16	146967.	146967.
17	74300.	74300.
18	144144.	144144.
19	144310.	144310.
20	39144.	39144.

## THE ELEMENT CONNECTIVITY MATRIX

1	1	5	2
2	5	6	2
3	6	3	2
4	6	7	3
5	7	4	3
6	7	8	4
7	9	6	5
8	9	10	6
9	10	7	6
10	10	11	7
11	11	8	7
12	11	12	8
13	13	10	9
14	13	14	10
15	14	11	10
16	14	15	11
17	15	12	11
18	15	16	12
19	17	14	13
20	17	18	14
21	18	15	14
22	18	19	15
23	19	16	15
24	19	20	16





THIS IS THE NUMSNP X 7 MATRIX

NODE	K2	K3	K4	K5	K6	K7
15.	0.	0.	0.	13.	14.	0.
96.	0.	15.	48.	81.	54.	0.
171.	0.	48.	114.	147.	90.	0.
141.	0.	80.	180.	108.	0.	0.
151.	18.	0.	0.	105.	152.	44.
458.	74.	148.	246.	279.	214.	95.
618.	134.	216.	312.	345.	276.	149.
345.	96.	284.	378.	216.	0.	0.
433.	98.	0.	0.	195.	338.	203.
938.	254.	352.	444.	477.	400.	257.
1098.	314.	420.	510.	543.	462.	311.
549.	192.	488.	576.	324.	0.	0.
709.	182.	0.	0.	285.	524.	365.
1418.	434.	556.	642.	675.	586.	419.
1578.	494.	624.	708.	741.	648.	473.
753.	288.	692.	774.	432.	0.	0.
585.	266.	0.	0.	0.	260.	527.
825.	614.	360.	0.	0.	286.	581.
903.	674.	396.	0.	0.	312.	635.
216.	384.	432.	0.	0.	0.	0.



# NODE NEIGHBOR CONNECTIVITY

NODE	K2	K3	K4	K5	K6	K7
1	0	0	0	5	2	0
2	0	1	5	6	3	0
3	0	2	6	7	4	0
4	0	3	7	8	0	0
5	1	0	0	9	6	2
6	2	5	9	10	7	3
7	3	6	10	11	8	4
8	4	7	11	12	0	0
9	5	0	0	13	10	6
10	6	9	13	14	11	7
11	7	10	14	15	12	8
12	8	11	15	16	0	0
13	9	0	0	17	14	10
14	10	13	17	18	15	11
15	11	14	18	19	16	12
16	12	15	19	20	0	0
17	13	0	0	0	18	14
18	14	17	0	0	19	15
19	15	18	0	0	20	16
20	16	19	0	0	0	0



[illegible]



## LIST OF REFERENCES

1. Nguyen, Dong H., "The Time-Dependent Nuclear Reactor with Feedback," Nuclear Science Engineering, v. 55, 1974, p. 307-319.
2. Salinas, D., Nguyen, D. and Southwarth, T., "Finite Element Solutions of a Nonlinear Reactor Dynamics Problem," Computational Methods in Nonlinear Mechanics, The Texas Institute for Computational Mechanics, 1974, p. 541-550.
3. Nguyen, D.H. and Salinas, D., "Finite Element Solutions of Space-Time Nonlinear Reactor Dynamics," Nuclear Science and Engineering, to appear.
4. Gear, C.W., "The Automatic Integration of Ordinary Differential Equations," Communications of the ACM, v. 14, March 1971, p. 176-179.
5. International Mathematical and Statistical Libraries Inc., (IMSL), v. I, 1975.
6. Felippa, Carlos A., Refined Finite Element Analysis of Linear and Nonlinear Two-Dimensional Structures, Report number SESM 66-22, Section I, University of California, Berkeley, October 1966.
7. Crank, J. and Nicolson, P., "A Practical Method for the Numerical Evaluation of the Solutions of Partial Differential Equations of the Heat-Conduction Type," Proceedings of the Cambridge Philosophical Society, v. 42, 1947 (England), p. 50-67.
8. Crandall, Engineering Analysis, McGraw-Hill Book Co., 1956, p. 39.





# INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, VA 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, CA 93940	2
3. Department Chairman, Code 59 Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93940	2
4. Associate Professor D.H. Nguyen, Code 59 Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93940	1
5. Associate Professor D. Salinas, Code 59 Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93940	1
6. LCDR Richard A. Olsen, USN USS SARATOGA (CV-60) FPO NEW YORK 09501	1







Thesis

0485

c.1

Olsen

Effective methods  
for solution of non-  
linear reactor dynam-  
ics problems using  
finite elements.

163522

Thesis

0485

c.1

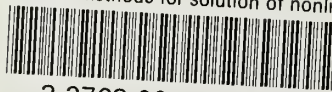
Olsen

Effective methods  
for solution of non-  
linear reactor dynam-  
ics problems using  
finite elements.

163522

thes0485

Effective methods for solution of nonlin



3 2768 001 97504 8

DUDLEY KNOX LIBRARY